Université de Toulouse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Institut National Polytechnique de Toulouse (INP Toulouse)

Présentée et soutenue par :
Warodom WERAPUN
Le jeudi 27 septembre 2012

Titre :

Architectures de réseaux pour la délivrance de services à domicile

---

ED MITT : Domaine STIC : Réseaux, Télécoms, Systèmes et Architecture

Unité de recherche :
UMR 5505

Directeur(s) de Thèse :

Mme. Béatrice PAILLASSA	Professeur INPT/ENSEEIHT
M. Julien FASSON	Maître de conférences INPT/ENSEEIHT

Rapporteurs :

M. Olivier DUGEON	Ingénieur de recherche Orange
Mme. Francine KRIEF	Professeur ENSEIRB

Autre(s) membre(s) du jury :

M. Michel DIAZ	Directeur de recherche du CNRS
M. Fabrice ARNAL	Ingénieur de recherche Thales Alenia Space
Mme. Béatrice PAILLASSA	Professeur INPT/ENSEEIHT
M. Julien FASSON	Maître de conférences INPT/ENSEEIHT

Architectures de réseaux pour la délivrance de services à domicile

Warodom WERAPUN

Thèse dirigée par

Béatrice PAILLASSA Professeur Directrice de thèse

Julien FASSON Maître de conférences Co-encadrant

Remerciements

Mes premiers remerciements iront à Béatrice PAILLASSA et Julien FASSON avec qui j'ai travaillé dès mon mastère et qui ont su me diriger vers le doctorat. J'aimerais leur adresser mes plus vifs remerciements pour leur dynamisme, leur tolérance, leur disponibilité, leurs compétences scientifiques et leurs exceptionnelles idées que j'ai pu apprécier tout au long de ma thèse. Ce travail n'aurait jamais pu aboutir sans eux, qui ont toujours su me consacrer des moments de leur temps, me témoigner leur soutien, me guider, et me conseiller.

Mes très vifs remerciements s'adressent aux membres du jury de thèse qui ont accepté de juger ce travail. Je suis profondément reconnaissant à mes deux rapporteurs, Monsieur Olivier DUGEON et Madame Francine KRIEF, qui ont bien voulu lire et évaluer ma thèse, de leurs remarques et analyse constructives. Je voudrais aussi exprimer ma gratitude à Monsieur Michel DIAZ pour avoir assuré la charge de président du jury. Je tiens également à remercier Monsieur Fabrice ARNAL qui m'a fait l'honneur d'accepter de participer au jury.

Merci au personnel du laboratoire, de l'INPT et de l'UPS pour les aides permanentes reçues. Je remercie en particulier Sylvie EICHEN et Sylvie ARMENGAUD qui ont été disponibles à chaque fois que j'ai eu besoin d'aide.

Je remercie également l'ensemble des enseignants et doctorants qui m'ont offert un excellent cadre de travail et ont rendu mon séjour extrêmement agréable. Merci à Anas Abou El KALAM qui m'a aidé durant le mastère, Anne Reynaud de MAZERAT qui m'a permis d'améliorer mon apprentissage de la langue française, et aussi Katia JAFFRES-RUNSER, André-Luc BEYLOT, Christian FRABOUL, Riadh DHAOU, Emmanuel CHAPUT, Jean-Luc SCHARBARG, Benoît ESCRIG, Jerôme ERMONT, Gentian JAKLLARI, Rahim KACIMI, Fabián ASTUDILLO et sa femme, Xiaoting LI, Mauricio ITURRALDE, Sokchenda SRENG, Nesrine BADACHE, Tony FLORES, Juan BARROS. Merci également à Chakadkit THAENCHAIKUN et sa femme qui m'ont aidé à faire un pot exceptionnel, merci aussi à Amadou Baba BAGAYOKO, Omer Landry NGUENA TIMO et Réjane DALCE qui m'ont guidé pour affronter les difficultés de la langue française.

Je souhaite également remercier mes amis: Aj. Charlie KREY, Girard LAURENT et P'Jeab, Ning et Ou-Jo, Teeka et N'Beau, M. Da@BaanSiam, P'Boon, P'Por et P'Shan, P'Suthira et Andy, K. Ji et Xavier, N'Yelly et Kai-Tong, N'Tuck, P'Ae, Den, Nop, P'Nui, P'Chon, P'Tom+, N'Pat et Sebastien, Pae, Joke et Sine, Nat et P'Aung, Khem et Yo, PJSoft, Matthieu, Evelin et Nicolas et tous ceux que je n'ai pas mentionnés ici.

Que mon université d’origine, Prince of Songkla University, CROUS et EGIDE (devenu Campus France) soient également remerciés pour leur aide.

Enfin et surtout, je remercie de tout cœur mes parents, ma femme, mon frère, ma sœur et mes bébés pour leur soutien sans faille. Ce grand succès n'aurait pu avoir lieu sans eux. Leur amour et encouragement sont les facteurs les plus importants de cet accomplissement.

Table des matières

Table des figures

Liste des tableaux

Acronymes

A.

AAA	Authentication Authorization and Accounting
ACK	Acknowledgement
AS	Application Server

C.

CAT	CATalogue server
CN	Correspondent Node
CP	Centralized P2P

D.

DHCP	Dynamic Host Configuration Protocol
DHT	Distributed Hash Table
DLNA	Digital Living Network Alliance
DMC	Digital Media Controller
DMP	Digital Media Player
DMPr	Digital Media Printer
DMR	Digital Media Renderer
DMS	Digital Media Server
DNS	Domain Name System
DP	DHT P2P
DSL	Digital Subscriber Line

G.

3GPP	3rd Generation Partnership Project
GENA	General Event Service Discover Protocol

H.

HGw	Home Gateway
HN	Home Network
HS	Home Service
HSS	Home Subscribe Server
HTTP	Hypertext Transfer Protocol

I.

I-CSCF	Interrogating Call Session Control Function
IBC	Identity Based Cryptography
IETF	Internet Engineering Task Force

IMS	IP Multimedia Subsystem
IPSec	IP Security
J.	
Jxta	Juxtapose
L.	
L2TP	Layer 2 Tunnelling Protocol
M.	
MPLS	Multi Protocol Label Switching
N.	
NGN	Next Generation Network
O.	
OSGi	Open Services Gateway initiative
P.	
P-CSCF	Proxy Call Session Control Function
P2P	Peer to Peer
PKG	Private Key Generator
PLC	Power Line Communication (or CPL)
PP	Pure P2P
R.	
RELOAD	REsource LOcation And Discovery
Q.	
QoS	Quality of Service
S.	
S-CSCF	Session Call Session Control Function
SDP	Session Description Protocol
SIP	Session Initial Protocol
SOAP	Simple Object Access Protocol
SN	Sender Node
SSDP	Simple Service Discovery Protocol
SSL	Secure Socket Layer
T.	
TCP	Transport Control Protocol
TISPAN	Telecommunications and Internet converged Services and Protocols for Advanced Networking
TLS	Transport Layer Security

U.

UA	User Agent
UDP	User Datagram Protocol
UE	User Equipment
UMTS	Universal Mobile Telecommunications System
UPnP	Universal Plug and Play
URI	Uniform Resource Identifier
URL	Uniform Resource Locator

V.

VoIP	Voice over IP
VPN	Virtual Private Network

X.

XML	eXtensible Markup Language

Introduction

Cette thèse s'est déroulée dans le cadre d'un projet d'intégration de technologies sur les réseaux d'opérateurs de services dans lequel il s'est agi de synthétiser, d'analyser et d'évaluer des technologies récentes.

Plus précisément les technologies dont nous avons traitées sont celles permettant de délivrer des services à domicile. L'objectif est de tirer avantage de l'intégration croissante des technologies de divertissement et de contrôle aux domiciles pour proposer de nouveaux usages de la communication.

La communication Internet s'enrichit de services à même de traiter les ressources de l'utilisateur. Dans le schéma conventionel actuellement déployé, le traitement des ressources s'effectue auprès des acteurs traditionels du service, dans cette thèse nous nous préocuppons d'un paradigme alternatif de traitement qui considère celui-ci au plus près de l'utilisateur. Le service devient une application de l'utilisateur qui repose sur une communication transparente, indépendante du contexte utilisateur. Les études présentées dans ce document ont pour sujet la mise en œuvre de cette communication.

Le schéma de transparence de la communication repose sur une communication inter-domicile sécurisée. L'utilisateur, qui n'est pas forcément en possession de l'équipement directement en charge de la gestion des ressources requise par un service, comme son téléphone portable ou bien sa tablette à son domicile, peut cependant accéder au service qui est par exemple la visualisation d'une vidéo qu'il a précédemment réalisée et stockée. Le service va traiter, récupérer puis transférer, les ressources dont l'utilisateur a les droits d'accès.

Dans cette thèse nous étudions les architectures de réseau aptes à délivrer des services aux domiciles des utilisateurs. Les architectures que nous étudions sont établies à partir d'éléments de technologie disponibles dont nous analysons les caractéristiques en vue de dégager leur possible adéquation. Plus précisément nous considérons les architectures de gestion de service d'une part dans le cadre des réseaux opérateurs en se reférant à l'architecture standard de gestion de service NGN, à savoir l'IMS, ainsi que d'autre part dans le cadre des acteurs de l'Internet en se référant aux architectures de pair à pair. Les schémas d'architectures que nous proposons et évaluons se soucient de fournir une communication sécurisée. Durant cette thése nous analysons l'impact de la sécurité sur différents schémas.

Notre méthode d'analyse repose sur la quantification de la signalisation induite par la délivrance de service, en spécifiant un environnement de référence constitué d'une architecture de service sécurisée de référence et d'un service de référence. La quantification est effectuée à l'aide d'expérimentations réelles nous permettant également de valider les concepts d'architectures que nous proposons ainsi que de simulations.

Principale contributions
Cette thèse s'inscrit dans une définition novatrice des services avec une gestion de ces derniers par des opérateurs réseaux et une réutilisation au mieux de technologies pérennes et fonctionnelles dont nous avons en premier lieu établi la cartographie. Une part de cette synthèse a été publiée dans l'ouvrage Service Management, In: Digital Home Networking, Romain Carbou, Michel Diaz, Ernesto Exposito, Rodrigo Roman (Eds.), ISTE - WILEY, 8, p. 259-306, 1, 2011). Ce travail bibliographique traite des architectures réseaux ainsi que de leur sécurité (une étude des vulnérabilités de la signalisation SIP a été publiée dans: Solution Analysis for SIP Security Threats, International Conference on Multimedia Computing and Systems, ICMCS 2009). L'étude met en exergue l'intérêt d'une gestion de ressource distribuée qui allie un contrôle d'authentification centralisé à même d'être supporté par un opérateur.

La thèse analyse plus précisément la gestion des services de partage de ressources. Une analyse de différents schémas est effectuée en considérant une localisation des ressources au domicile alors que la localisation des index des ressources est soit centralisée dans le réseau, soit distribuée chez les utilisateurs. La première contribution de cette analyse est la proposition d'une architecture de gestion sécurisée reposant sur une signalisation SIP qui offre une alternative interessante à l'architecture de référence du NGN, L'IMS. L'intérêt de l'architecture est évalué au travers de l'étude comparative de signalisation à plusieurs niveaux de sécurité et la réalisation d'expérimentation (Publié dans: Network Architecture for home service delivery, The fourth International Conference on mobile ubiquitous computing, system, services and technologies, UBICOMM 2010).

La deuxième contribution architecturale porte sur la mise en œuvre d'une architecture reposant sur des schémas de pair à pair. L'aspect innovant de notre proposition porte sur l'intégration d'un aspect opérationnel à même de sécuriser la gestion des ressources, dans les architectures traditionnelles de pair à pair déployées sur Internet. Pour étudier l'adéquation des scénarios d'architecture nous avons élaboré une modélisation originale du coût de signalisation. Celle-ci permet de déterminer les facteurs de décision en faveur d'une architecture ou d'une autre selon (Publiée dans: Home Service Communities and Authentication, Internal Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11). Par ailleurs nous proposons une preuve de ce concept sur un schéma en voie de standardisation, RELOAD.

Organisation de la thèse
La présentation des travaux s'organise en 5 chapitres (Figure 1) que nous synthétisons de la façon suivante :

Chapitre 1 – Services et réseaux à domicile : Ce chapitre explicite les notions sur lesquelles s'appuie la thèse. Il présente les caractéristiques des services et des réseaux domestiques, ainsi que des architectures d'interconnexion entre domiciles. La présentation d'une architecture et d'un service de référence qui permettront d'évaluer les propositions d'architectures détaillées dans les chapitres ultérieurs conclu ce chapitre

Chapitre 2 – Mécanismes d'authentifications : Une synthèse non exhaustive des mécanismes de sécurité est présentée en suivant le point de vue de l'architecture réseau. L'objectif est de se concentrer sur les aspects d'échanges d'informations, c'est-à-dire la signalisation de mécanismes existants.

Chapitre 3 - Architectures centralisées pour la délivrance de services au domicile : ce chapitre propose plusieurs méthodes d'authentification avec une signalisation SIP, qu'il compare avant de présenter un choix d'architecture, SIP proxy. La signalisation, tant des mécanismes que des architectures, est étudiée en référence à des expérimentations et documents de normes.

Chapitre 4 - Architectures distribuées pour la délivrance de services au domicile : Une étude des architectures P2Ps distribuées est menée et une comparaison avec les architectures centralisée étudiées dans le chapitre précédent est proposée. Deux architectures P2P sont analysées à savoir une architecture de « pur » P2P avec un algorithme de recherche par innondation, et une architecture DHT, avec un algorithme de recherche sur réseau structuré. La comparaison des architectures sécurisées s'appuye sur l'expérimentation et la simulation de réseaux pairs à pairs.

Chapitre 5 – Conclusion et perspectives : un bilan de travaux menés sur les trois architectures et l'influence du choix du scénario de référence sur notre travail est abordé. Les interactions entre opérateurs de services sont également indiquées en perspective d'étude.

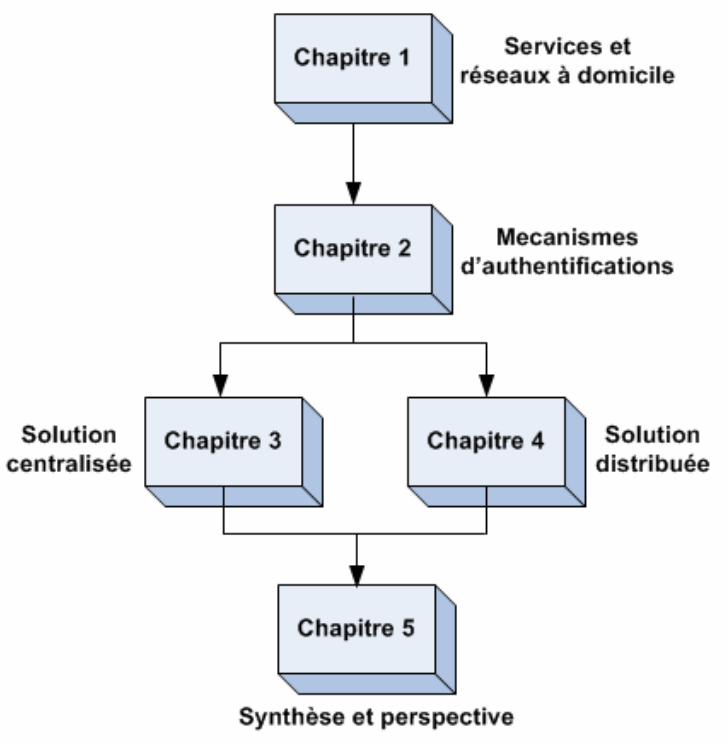

Figure 1: Organisation de la thèse

Chapitre 1

1. Services et réseaux à domicile

Sommaire

1.1. Introduction

Dans cette étude nous nous intéressons à un cadre de services émergeant, les services à domicile, autrefois qualifiés de domotique. Cette étude cible donc ces services que nous qualifierons pour toute la suite du document, Home Services (notés HS). Les HS sont déployés dans un cadre communiquant à domicile. Ce réseau est appelé Home Network (HN) et dispose d'une passerelle régissant ce réseau et ses interactions avec le monde extérieur dénommée Home Gateway (HGw).

Ce chapitre précise le contexte de notre étude. Dans un premier temps nous présentons le projet à l'origine de ce travail : Feel@Home. Nous nous intéresserons dans un second temps aux deux concepts clefs de ce travail ; les notions de service à domicile (HS) (section 1.3) et de réseau domestique (HN) (section 1.4). Après avoir illustré ces deux notions, une typologie des services permettra de mettre en relief les différents aspects des services, leur besoin et leurs orientations (section 1.5). Puis, nous aborderons les aspects communicants nécessaires à la mise en œuvre des services (section 1.6). Nous conclurons le chapitre en explicitant le service qui servira de référence comparative aux propositions d'architectures de gestion de service que nous détaillerons dans les chapitres suivants.

1.2. Feel@Home

Le travail de cette thèse prend appui sur le projet Feel@home, auquel nous avons participé. Feel@Home est un projet Européen CELTIC regroupant des opérateurs téléphoniques, des universitaires et des sociétés de services. Le projet s'est déroulé de 2008 à 2010. Le principal objectif était d'introduire de nouveaux services à domicile, à savoir des services multimédia et de contrôle d'équipements. Feel@Home vise à une adoption massive par les utilisateurs des « Digital Home advanced audiovisual networked services » dans la lignée des travaux menés sur l'automatisation du domicile et sur l'extension de ceux-ci via le réseau.

Pour valider le concept et démontrer ses avantages, Feel@Home a développé une architecture ouverte basée sur les standards de facto. Cette architecture a été déployée avec le matériel requis et les composants logiciels et testée sur plusieurs sites de démonstration interconnectés situés dans différents pays européens.

1.2.1. Les objectifs du projet

Le propos de Feel@Home est la gestion de contenu numérique, la gestion automatique du réseau au domicile (home area network) et le traitement des offres de services multi-utilisateurs. En outre, il fournit un nouveau paradigme permettant une interaction transparente, une prestation de services personnalisés et sensibles au contexte. Les utilisateurs doivent être capables d'accéder à leurs services via des dispositifs divers, et cela peu importe leur localisation : à domicile, chez un autre abonné Feel@Home ou distant. De plus, au delà de l'accès à son réseau, Feel@Home se propose d'offrir à l'utilisateur l'accès à d'autres réseaux de particuliers, sous réserve qu'ils lui soient ouverts.

Les technologies utilisées dans le projet sont les technologies standards : UPnP/DLNA ainsi que celles afférentes à l'Internet au sens large (telles SIP, HTTP, IMS..).

L'accès à Internet, en raccordant indirectement les domiciles des utilisateurs, doit permettre l'accès extérieur aux différents services et informations de manière fiable : le projet a donc défini un cadre pour la qualité de services (QoS) de bout en bout et la sécurité. Feel@Home assure l'authentification fiable et transparente des utilisateurs, la confidentialité et une expérience de qualité pour les consommateurs et les fournisseurs des services numériques à domicile.

1.2.2. Travail effectué dans le projet

Dans le projet Feel@Home différents thèmes d'études ont ainsi été entrepris, à savoir :

- la définition des scénarios de service. Les services qui ont été analysés dans le projet sont soit des services de domotique, soit des services de partage plutôt orienté communautaires. Alors que les premiers s'exécutent au domicile via un accès au domicile local ou distant, les seconds s'exécutent entre les membres d'une communauté. Ce réseau social est du domaine de la sphère privée et donc fermé entre les domiciles équipés de box Feel@Home;
- le cadre général du travail (caractérisation des services, méthodes de mise en œuvre) ;
- la personnalisation des contextes ;
- la sécurité ;
- l'architecture réseau opérateur ;
- les équipements.

Notre contribution qui sera présentée pour partie dans cette thèse a consisté à l'étude des architectures réseau et de la sécurité au travers de la rédaction de différents rapports et de la réalisation de démonstrations [PaFW11][WFPM10][WFPM09]. D'autres aspects concernant l'architecture réseau ont conduit à une étude sur la qualité de service, dont les contributions peuvent être trouvées dans la thèse de [MaDD11].

1.3. Services à domicile

Précisons en premier lieu la notion de services dans le contexte du domicile avant de s'intéresser à leur mise en œuvre.

1.3.1. Notion de domicile automatisé

Il s'agit d'automatiser l'activité sur le lieu de vie de l'utilisateur nommé domicile (par opposition au lieu professionnel) en vue d'assurer un meilleur confort pour l'usager, ou encore d'augmenter l'efficacité énergétique et la sécurité du domicile. La domotique, relative à l'utilisation d'un domicile automatisé, est le processus d'accès et de contrôle des appareils domestiques à partir d'emplacements distants via Internet ou

un réseau domestique. Des exemples de contrôle d'appareils sont la mise en œuvre d'un contrôle centralisé de l'éclairage, du chauffage ou encore de la climatisation. La domotique est également étudiée dans le cadre des personnes dépendantes, en vue d'en améliorer la qualité de vie en facilitant leur maintien au domicile et en minimisant le recours aux personnels et soins médicaux institutionnels.
Les techniques employées dans l'automatisation de la maison comprennent celles relatives aux automatismes du bâtiment ainsi que celles du contrôle des activités domestiques, du divertissement à l'entretien.

Plus précisément, voici une illustration des tâches qui peuvent être menées à bien par de la domotique associée aux principaux automatismes :

- **CVC :** Chauffage, Ventilation et Climatisation (CVC) pour le contrôle de la température et de l'humidité. Un thermostat contrôlé par Internet peut à la fois être source d'économies et préserver l'environnement (par exemple une personne en voyage et dont le retour n'est pas connue au moment de son départ, peut arrêter son chauffage et le reprogrammer un peu avant de revenir chez elle).
- **Eclairage :** avec par exemple, l'extinction des lumières de la maison sur une plage de temps à une heure et date prédéterminée, ou lorsqu'il n'y a plus d'occupants dans une pièce. Il peut également s'agir d'allumer les lumières, de contrôler la luminosité de l'éclairage en fonction du niveau de lumière ambiante disponible, ou encore de changer la couleur ambiante d'une pièce (contrôle de l'humeur).
- **Audio et vidéo :** pour le contrôle des entrées et des sorties des équipements de divertissement (TV, vidéo, console de jeux, tablettes…). De multiples sources audio ou vidéo peuvent être sélectionnées et distribuées sur un ou plusieurs équipements (*renderers*).
- **Sécurité :** l'utilisateur peut sélectionner et regarder en direct des caméras installées à son domicile depuis un point de raccordement Internet ; celles-ci peuvent également être contrôlées, pour permettre à l'utilisateur d'observer l'activité autour de son habitation. Par ailleurs, les systèmes de sécurité peuvent inclure des détecteurs de mouvement aptes à notifier l'utilisateur via le système de sécurité ou par un téléphone cellulaire de tout mouvement non prévu. Un autre exemple de tâche à but sécuritaire est le contrôle de l'accès au domicile par interphone qui est intégré à un autre dispositif tel un système téléphonique ou un téléviseur pour, par exemple, visualiser sur l'écran de télévision la porte d'entrée du domicile.
- **Robotique :** il s'agit de contrôler les robots domestiques, à partir du réseau domotique ou de l'Internet.

1.3.2. Notion de service à domicile

La notion de service varie selon qu'elle est relative à l'opérateur réseau qui propose un service ou à l'utilisateur qui s'en sert. La notion de service au sens opérateur NGN (*Next Generation Networks*) est historiquement relative aux transports d'informations tels que les appels vocaux, les flux vidéo et les services web. La plupart de ces services sont déployés sur les serveurs de l'opérateur et sont administrés par ce dernier, en particulier via l'architecture d'administration de service IMS normalisée dans le cadre des opérateurs NGN. Celle-ci est chargée de permettre l'accès d'un

utilisateur à ses services quelle que soit sa localisation ; il s'agit de gérer des possibilités d'accès à des services : le niveau de qualité de transfert pour un service, les droits d'utilisations, le type d'équipement/d'accès utilisé…

Pour l'utilisateur, la vision d'un service est très différente puisqu'il s'agit d'un ensemble de possibilités qu'on lui offre. Dans les communications les services étaient à l'origine assez basiques (la télévision, la radio et le téléphone). Puis vint l'accès à d'autres types de services via TRANSPAC (le minitel) et enfin Internet. Aujourd'hui, presque deux milliards et demi des personnes y ont accès, souvent via une passerelle à leur domicile dénommée box. Les services classiques étaient plutôt statiques dans le sens où l'utilisateur avait très peu d'interactions, le service lui offrant une variété d'actions limitées (choisir la chaine pour la télévision, choisir l'interlocuteur pour le téléphone). Il s'agit alors pour l'opérateur d'un simple flux de données à traiter de manière uniforme avec les autres flux du même type. Toutefois, l'évolution du web dynamique (web 2.0) a amené d'autres types de services pour l'utilisateur. Au-delà du transport de l'information, il requiert également le traitement de celle-ci. Des nouveaux besoins apparaissent avec notamment les réseaux sociaux qui demandent à la fois une nouvelle structuration via des métadonnées (données qui explicitent les données) ou aussi la gestion des identités comme avec Facebook, Blogger, Picasa ou YouTube. Contrairement aux services classiques, nous qualifions ces services de dynamiques car ils offrent une plus grande interactivité aux utilisateurs. Ils sont plus adaptifs que les services traditionnels en autorisant l'utilisateur non seulement à démarrer ou arrêter le service, mais aussi à créer et modifier sa configuration. Le flux de données est alors très différent et demande à l'opérateur et/ou au fournisseur de services une gestion particulière. Les données, elles-mêmes, sont généralement gérées et stockées par l'opérateur du service sur ses serveurs (organisés en nuage, *cloud*).

Toutefois cette gestion par l'opérateur implique deux ambigüités : l'opérateur devient propriétaire des données personnelles de l'utilisateur (les photos, les mails ou encore les documents de travail) et il se pose alors la question de la confidentialité. D'autre part, puisque l'opérateur devient le dépositaire de ces données, sa responsabilité est engagée notamment lorsqu'il s'agit de contenu illicite, posant alors la question de sa neutralité. Une autre solution pour ce type de service consiste à ne pas avoir les données gérées par le fournisseur de service (par exemple Facebook), mais directement chez l'utilisateur. Dans ce cas, l'accès serait géré par l'opérateur de réseau mais le stockage resterait chez le client et donc sous sa propriété et sa responsabilité. Nous considérons que cette configuration correspond à un service en mode domicile par opposition à un mode cloud. Le rôle de l'opérateur de réseau dans ce contexte est alors d'assurer une qualité d'accès au service satisfaisante avec en particulier le déploiement et la gestion de certains mécanismes d'authentification avant de laisser l'utilisateur prendre en charge ces fonctions.

Dans cette thèse nous nous intéressons au déploiement de services dynamiques qui s'exécutent dans un mode de service à domicile. Les utilisateurs sont gestionnaires de leurs propres données. Un même utilisateur peut accéder à ses services via divers moyens de connexion et divers appareils. En outre, ces services à domicile peuvent

être déployés par un opérateur réseau ou directement par l'utilisateur. Ils peuvent être contrôlés localement et à distance.

Par ailleurs, pour s'exécuter, les services peuvent requérir des communications complémentaires : citons par exemple le cas d'un service qui s'appuie sur des informations météorologiques issues de capteurs. Il s'agit alors d'élaborer un service composé de services plus élémentaires. Cette notion rejoint le concept de « web services ». Cependant alors que les web services sont majoritairement fournis par un serveur, dans le cas des services à domiciles, comme nous l'avons précisé, l'exécution a lieu au domicile avec un élément client et un élément serveur et elle peut également être étendue avec un élément serveur externe au domicile en mettant en place une solution de connectivité hors domicile.

Les services à domicile ont fait et font l'objet de nombreuses recherches que ce soit dans le domaine de l'informatique « pervasive », de la santé ou encore de l'énergie. Une vision des services à domicile reposant sur l'informatique ubiquitaire est exposée en 1991 par [Weis99]. Les principaux composants technologiques identifiés par l'auteur sont des ordinateurs bas coûts disposant d'affichage adaptés (écrans tactiles, tablettes, écrans plats..) des logiciels pour des applications ubiquitaires et un réseau liant l'ensemble. Vingt ans après, les ordinateurs bas coûts de type smartphone et autres sont disponibles, de même que les middlewares avec des standards comme OSGI, ainsi que des systèmes de communications standards au domicile, tels UPnP et DLNA que nous détaillerons par la suite. Par rapport à ces travaux pionniers relevant surtout de l'automatisation du domicile, une dimension nouvelle apparaît : l'interconnexion 'Internet'. Avec elle, les services ont pris un tournant décisif vers la vie privée d'une part mais aussi vers les préoccupations de la société entrée dans le 21ème siècle (la maintien au domicile des personnes âgées [NFFT08] ou la gestion de l'énergie).

De nombreux programmes de recherche sur l'automatisation du domicile ont été menés comme au Japon avec l'e-japan strategy project en 2001 [EJAP01] et l'e-life initiative en 2003 [METI03], ou encore en Europe avec les programmes européens ITEA1 et ITEA2, (Information Technology for European Advancement) en 2005. Un exemple de ces services est indiqué dans [YHLO03], il y est propose le développement d'un service Accueil Robots «ISSAC» : ce robot est conçu pour le nettoyage mais aussi la détection d'intrus avec un transfert des images capturées (à la police par exemple). Plus récemment, [LePK06] envisage le déploiement des services numériques à domicile avec une plateforme de gestion pour gérer les passerelles domestiques et le cycle de vie des services. L'objectif est d'avoir un système ouvert (reposant sur une technologie OSGI) apte à supporter des services variés. La plateforme de service a vocation à être installée sur la passerelle de l'utilisateur et être administrée et mise à jour par un opérateur. Il est envisagé que les fournisseurs de services enregistrent des applications développées pour la plate-forme et que celles-ci soient téléchargées sur les passerelles résidentielles. [Anan07] s'intéresse au déploiement de service multimédia avec une architecture DLNA et une communication distante par IMS. Les utilisateurs peuvent choisir dynamiquement différents services pour le partage de contenus numériques sans aucune configuration

complexe, grâce à la passerelle IMS. Différents cas d'utilisations sont indiqués : le cryptage de contenu de données, le changement du réseau au domicile vers le réseau cellulaire, le retrait de données situées au domicile à partir d'un terminal mobile. [MFMA10] propose d'utiliser une architecture en P2P pour le déploiement de services sur les passerelles. Les auteurs explorent comment les technologies P2P JXTA peuvent être utilisées pour mettre en œuvre un service idoine qui permette aux périphériques situés à différents endroits d'être combinés dans une configuration personnalisée.

1.4. Réseau domestique

Les Home Networks, HN, (réseaux situés au domicile de l'utilisateur) proposent une automatisation du domicile qui repose sur un ensemble de technologies standardisées que nous présentons dans cette section

1.4.1. Notion de Home Network

La mise en œuvre des HSs repose sur les capacités de communications des divers dispositifs résidant au domicile (CVC, audio, vidéo…), mais aussi sur leur accessibilité hors du domicile. Il est nécessaire non seulement d'atteindre les appareils domestiques que l'on cherche à contrôler mais aussi de pouvoir se connecter via des dispositifs extérieurs au domicile, qui peuvent servir de contrôleur voire de « renderer » (écoute de musique, lecture, visionnage de films) par leur capacité à traiter des documents multimédia.

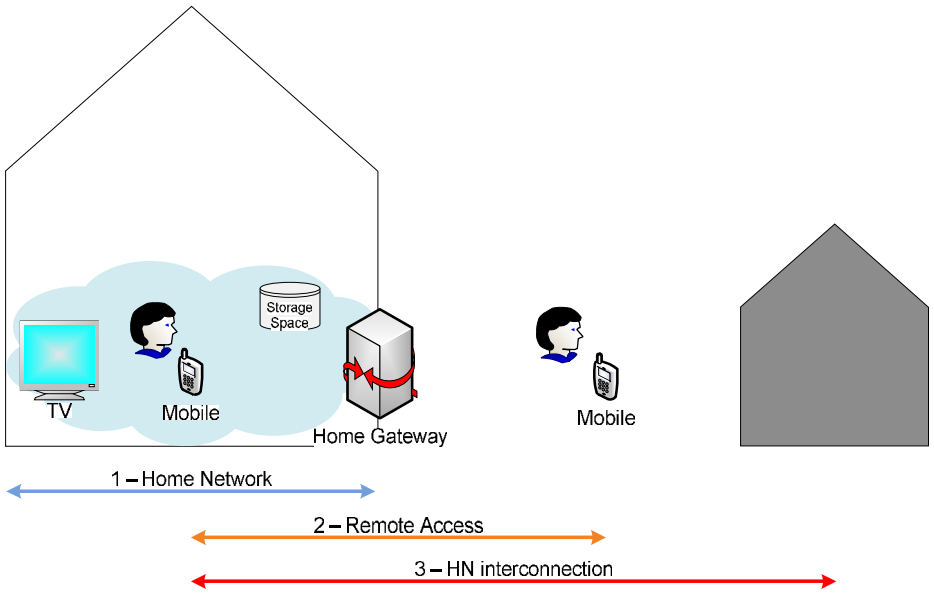

Figure 1.4.1-1**:** Les trois types de communication d'un Home Network

Il y a alors trois aspects différents dans la notion de Home Network (Figure 1.4.1-1) :

- La communication entre les équipements du domicile (1). C'est là l'objet du HN : interconnecter des équipements proposant des interfaces différentes, comme par exemple du Wifi, du Zigbee, de l'Ethernet ou encore du CPL.
- La communication avec des équipements en dehors du domicile, qui nécessite l'intermédiaire d'une passerelle dénommée Home Gateway (2). Il s'agit là de permettre à un utilisateur du HN de pouvoir s'interconnecter à distance à ses HSs et ainsi les contrôler.
- L'interconnexion avec d'autres domiciles (3). Ce point sera l'objet principal de notre travail au travers de l'étude des architectures réseaux permettant sa mise en œuvre.

Figure 1.4.1-2: Un exemple de réseau à domicile

Cette partie s'intéresse uniquement au HN en lui-même, c'est-à-dire aux technologies permettant de gérer un réseau tel que celui présenté dans la Figure 1.4.1-2. L'utilisation de différentes interfaces pour une mise en réseau des équipements domestiques nécessite de définir une architecture de communication permettant l'échange de données le plus simplement possible, c'est-à-dire sans avoir recours à une compétence 'informatique' de l'utilisateur d'une part et, d'autre part, sans demander beaucoup des équipements terminaux qui peuvent avoir de faibles capacités de traitements. Ainsi, le réseau se doit d'être transparent. Cette caractéristique de transparence a été traité dès 1999 par des groupes de standardisation : Universal Plug and Play *(UPnP)* forum pour la partie applicative et le groupe IETF Zero Configuration Networking *(Zeroconf)* pour la partie réseau.

La suite de cette section présente deux standards complémentaires mettant en œuvre la transparence pour les HNs, uPnP et DLNA.

1.4.2. UPnP

1.4.2.1. Les bases d'UPnP

La technologie Universal Plug and Play est promue par le Forum UPnP [UPnP] à travers plusieurs comités chargés entre autre de la qualité de service des équipements audio vidéo et du raccordement à Internet (uPnP). Le concept de l'UPnP est à la base une extension du Plug-and-Play, une utilisation directe des périphériques sans

installation ou configuration extérieure. Les périphériques UPnP lorsqu'ils se connectent à un réseau établissent automatiquement les configurations nécessaires pour travailler avec d'autres appareils. Pour cela UPnP s'appuie sur des technologies Web qui permettent aux différents équipements du domicile d'interagir. Plus précisément UPnP propose un ensemble de protocoles permettant à chaque équipement de découvrir les autres. Le réseau ainsi construit offre des services de partage des données, de communication et de divertissement.

UPnP [UPnP08] définit l'architecture de communication entre équipements et points de contrôle. Un point de contrôle fonctionne comme un serveur répondant à des requêtes émises par un équipement. L'architecture a pour objet de créer un environnement ouvert pour la mise en œuvre de services entre les périphériques en utilisant des technologies courantes telles que TCP, UDP, SOAP et XML. Au niveau le plus élevé de l'architecture, les messages contiennent des informations UPnP spécifiques aux services. Ces informations sont complétées au niveau inférieur par des informations définies par UPnP. Les informations issues des niveaux inférieurs de l'architecture sont traitées par des protocoles spécifiques tels Simple Service Discovery Protocol (SSDP) pour la découverte de service ou General Event Notification Architecture (GENA) pour la gestion d'évènement. Les messages SSDP sont transmis en multicast ou bien en UDP unicast. Les messages GENA, sont quant à eux transférés par HTTP. Tous ces messages sont ensuite encapsulés dans des paquets IP (V4 ou V6).

L'architecture UPnP supporte en partie des solutions issues d'un groupe de travail de l'IETF pour mettre en œuvre l'objectif de zéro configuration réseau (groupe zeroconf). Un dispositif compatible UPnP de n'importe quel fournisseur peut rejoindre dynamiquement un réseau, obtenir une adresse IP, annoncer son nom, transmettre ses capacités à la demande, et se renseigner sur la présence et les capacités des autres périphériques. Les serveurs de configuration d'adresse, Dynamic Host Configuration Protocol (DHCP), et de nom, Domain Name System (DNS) sont optionnels et ne sont utilisés que s'ils sont disponibles sur le réseau (L'utilisateur n'a pas à savoir installer un serveur pour que le réseau fonctionne). Les périphériques peuvent se déconnecter du réseau automatiquement sans laisser des informations d'état.

1.4.2.2. Fonctionnalités élémentaires

Les éléments de base de la mise en réseau automatique sont l'adressage, la découverte, le contrôle, la gestion des évènementiels, et la présentation.

Adressage
L'adressage est un adressage IP. Chaque dispositif doit mettre en œuvre un client DHCP et rechercher un serveur DHCP lorsque l'appareil se connecte au réseau. Si aucun serveur DHCP n'est disponible, l'appareil doit s'attribuer une adresse. L'adresse est une adresse locale au domicile qui n'est pas routable ni enregistrable sur le DNS. Elle est de la forme 169.254.0.0/16 et est choisie aléatoirement par l'équipement qui vérifie ensuite l'unicité de l'adresse choisie sur le réseau [ChAb05].

Découverte
La découverte de service a été étudiée par l'IETF (via soit le Service Location Protocol, soit un mécanisme reposant sur le multicast DNS : m-DNS) ; cependant le protocole commercialement installé (en particulier par Microsoft) est le protocole SSDP qui a été choisi par UPnP. Quand un périphérique est ajouté au réseau, SSDP annonce ses services aux points de contrôle sur le réseau, par une annonce cyclique en multicast. De même, quand un point de contrôle est ajouté au réseau, SSDP permet à ce point de contrôle de rechercher des périphériques. Dans les deux cas il y a un échange de messages de découverte contenant des précisions essentielles sur l'appareil ou l'un de ses services, comme par exemple, son type, identifiant, et un pointeur vers des informations plus détaillées (URL).

Description
Lorsqu'un point de contrôle a découvert un équipement/service, il récupère par un demande HTTP Get adressée à l'URL fournie dans le message de découverte reçu, une description UPnP en XML ainsi que des informations constructeurs (type, nom du fabriquant). La description des services UPnP contient une liste de commandes (actions) auxquelles répond l'équipement/service avec les arguments associés. Une liste de variable permet également de décrire les états du service en cours d'éxécution en termes de types de données, de plages de valeurs et d'évènements.

Contrôle
Le point de contrôle émet des actions vers le point contrôlé (connues via la phase précédente de description) à l'adresse URL du service. Les messages sont exprimés en XML via le protocole Simple Object Access Protocol (SOAP). En réponse aux actions le service change d'état conformément à sa description (changement de valeurs de variables…)

Evénementiel (*eventing*)
Le protocole de notification d'événements défini dans l'architecture UPnP est « General Event Notification Architecture » (GENA). Lorsqu'un service est modifié il publie les mises à jour via l'envoi de messages évènements en XML (*Gena notify*). Les points de contrôle abonnés à la publication de cet évènement reçoivent ces messages (à l'initialisation un message d'abonnement est émis par un point de contrôle, *GENA subscribe*). Les évènements peuvent aussi être reçus sans abonnement explicite via un service multicast (non fiable reposant sur UDP).

Présentation
C'est la dernière étape de la mise en réseau UPnP. Le point de contrôle peut récupérer à partir d'une URL une page de présentation, qui sera accessible à l'utilisateur via son chargement dans un navigateur. Il a ainsi la possibilité à partir de cette interface de contrôler l'équipement ou le service.

1.4.2.3. UPnP Audio Video

L'architecture de base est complétée pour les équipements audio video (lecteur Blu-ray, disque dur externe, appareil photo, téléphone mobile, téléviseur, ordinateur,

tablette…) par l'architecture UPnP Av. Celle-ci définit deux types d'équipements: le *media server*, qui héberge le contenu multimédia qui va être parcouru par le point de contrôle et le *media renderer* qui est l'équipement qui exécute (affiche/sonorise) le contenu.

Le logiciel UPnP AV est souvent installé sur une box (comme par exemple sur la Freebox) et il permet de se servir de son écran de télévision pour accéder à des équipements locaux.

1.4.3. DLNA

1.4.3.1. Principes de DLNA

Digital Living Network Alliance (DLNA) [DLNA] est un organisme créé en 2003 et regroupant pas moins de 250 acteurs du monde du multimédia (essentiellement des constructeurs d'appareils électroniques et multimédia ainsi que des fournisseurs de service). L'objectif du DLNA rejoint celui d'UPnP en termes d'interopérabilités entre les équipements de constructeurs différents. Il s'agit toutefois plus d'un service que d'une gestion de réseau : DLNA offre un service de partage de photos numériques, musiques et vidéo via un mode client/serveur avec un format figé. L'exemple le plus classique est l'utilisation d'un téléviseur DLNA et d'un serveur DLNA sur un PC pour lire des données multimédia (photos, musiques, ou vidéos) du PC vers la télévision (Figure 1.4.3-1). La certification DLNA est de plus en plus répandue chez les équipementiers (TV, console, …) et tout PC avec une interface réseau (par exemple Ethernet ou WiFi) peut en installant un logiciel certifié devenir un dispositif DLNA. Il est estimé que plus de 440 millions de dispositifs certifié DLNA sont actuellement installés aux domiciles des utilisateurs.

Figure 1.4.3-1: Exemple d'interopérabilité des équipements certifiés DLNA [DLNA]

Un client DLNA accède à un contenu multimédia DLNA grâce à une mise en réseau UPnP. Le client DLNA détecte les serveurs DLNA disponibles et peut alors effectuer des recherches sur les contenus, puis les lire à distance en suivant la norme.

1.4.3.2. Fonctionnalités du DLNA

Le DLNA définit les fonctionnalités de son service au travers de différentes classes d'équipements représentées sur la figure suivante (Figure 1.4.3-2).

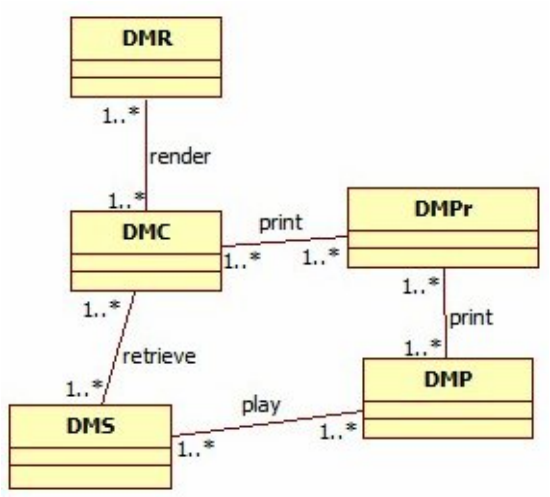

Figure 1.4.3-2: Les interactions entre les différentes classes d'équipements du DLNA

Les différentes classes proposées sont :

- Digital Media Server (DMS) : Cet appareil stocke les contenus accessibles à des clients, des DMP ou des DMR (Exemples : PC, disques de stockage externe).
- Digital Media Player (DMP) : Ces dispositifs recherchent du contenu sur des DMS et offrent la possibilité de les lire (Exemple : téléviseurs, chaînes stéréo).
- Digital Media Renderer (DMR) : Un DMR est équivalent à un DMP dans sa fonctionnalité de rendu, toutefois l'équipement est incapable de trouver des données par lui-même (Exemples : téléviseurs, haut-parleurs sans-fils). La recherche, la sélection et le contrôle du service se fait alors par une autre classe d'équipement, le DMC.
- Digital Media Controller (DMC) : voir ci-dessus (Exemples : tablettes, smatphones).
- Digital Media Printer (DMPr): Ces appareils offrent des services d'impression au réseau domestique DLNA. (Exemple : imprimantes photo en réseau)

1.4.3.3. Lien entre DLNA et UPnP

Même si UPnP et DLNA sont normalisés par deux instances différentes, la certification du DLNA est dérivée de l'UPnP. DLNA propose un cadre plus restrictif, réduisant les formats de présentation des contenus, les formats des données multimédia, etc… Cette délimitation plus contraignante a pour objectif une meilleure interopérabilité entre les équipements du HN. Ainsi, si un client UPnP peut fonctionner à partir d'un serveur DLNA sans problème, il est possible qu'un serveur

UPnP ne puisse interagir avec un périphérique DLNA s'il lui fournit par exemple une donnée dans un format non reconnu.

On pourra ainsi retenir qu'UPnP est plus ouvert que DLNA au risque d'avoir des problèmes de compatibilité entre équipements.

1.5. Classification des services à domicile

Après avoir présenté les notions de HS et de HN, il s'agit de distinguer les principales caractéristiques des services à domicile. Quelles fonctionnalités doivent être mises en œuvre et par qui ? Classiquement, les rôles sont partagés entre l'utilisateur, l'opérateur réseau et le fournisseur de services. Toutefois, pour un accès de type privé (aussi dénommé accès domestique), l'opérateur réseau est aussi un fournisseur de services, dans la simple mesure où il offre déjà un accès à Internet. De plus ce dernier via sa *box* propose un certain nombre d'autres services comme le bien connu Triple Play. Une évolution est alors de fournir des services à domicile plus variés via la *box* devenant une Home Gateway (HGw). Par la suite nous employerons le terme opérateur réseau même s'il œuvre aussi comme un fournisseur de services.

La spécification des services passe par une classification selon différents critères ; ici nous considérons le mode de fonctionnement, la localisation du service, le mode d'accès et le type d'informations partagées.

1.5.1. Mode de fonctionnement

Nous distinguons deux modes de fonctionnement : le mode statique et le mode dynamique.

Le service statique est un HS stable et simple. L'utilisateur peut uniquement démarrer et arrêter le service. La configuration du service a été effectuée par l'opérateur du réseau, par exemple, le « *Triple Play* » d'une *box* Internet ou son fournisseur de services, si celui-ci est différent.

Le service dynamique est plus adaptatif. Non seulement l'utilisateur peut démarrer ou arrêter le service, mais il peut le configurer via le réseau domestique. L'opérateur du réseau peut dans un premier temps déployer et gérer certains mécanismes d'authentification mais il laisse une grande autonomie à l'utilisateur. Ces services peuvent être du type contrôle du domicile, comme des services de réseaux communautaires inter-domiciles.

1.5.2. Localisation et accès au service

Les services peuvent être classés par leur cadre d'exécution, notamment en termes de localisation du service, accès par l'utilisateur et besoin de communications. Bien sûr chacun de ses points peut avoir un impact sur les autres ; par exemple un service déployé sur un serveur sur Internet nécessite un accès distant de la part de l'utilisateur.

1.5.2.1. Localisation du service

La figure suivante (Figure 1.5.2-1) présente les différents emplacements où un service peut être déployé.

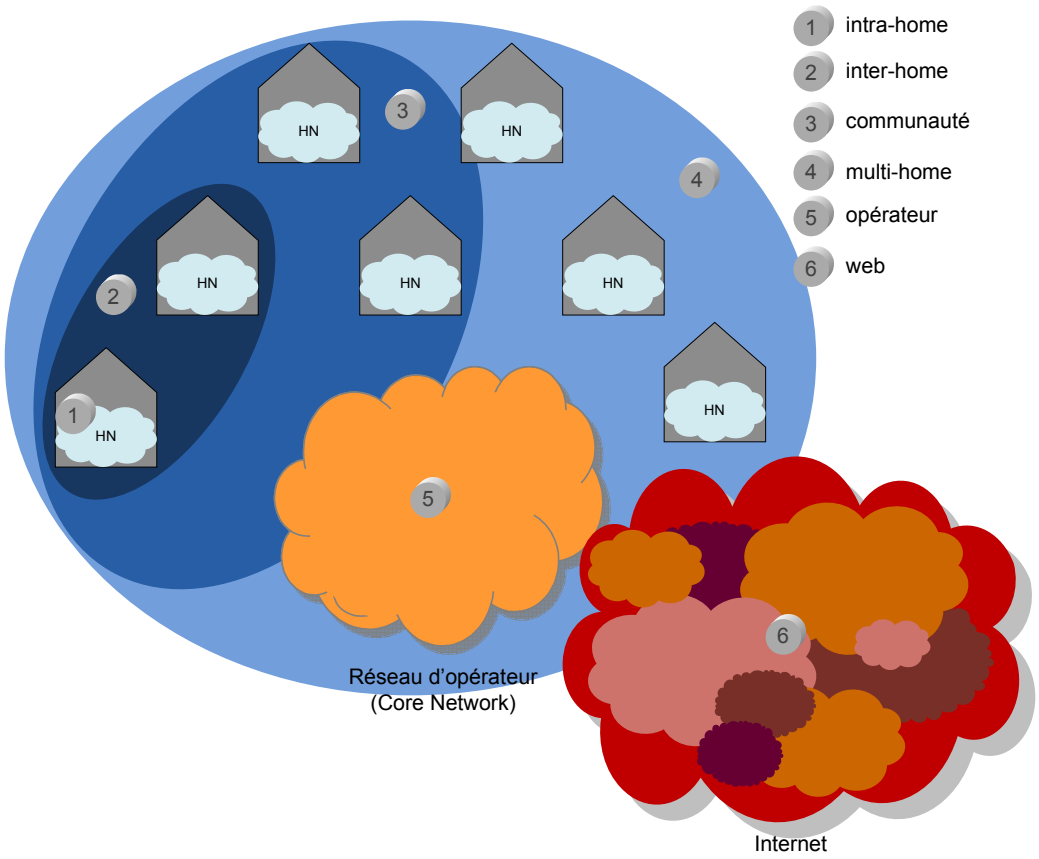

Figure 1.5.2-1: Localisation du service

Le premier cadre de déploiement est le domicile (1). Pour ces services, il n'y a pas de recherche de ressources à effectuer en sus de celle qui est mise en œuvre par le HN. Cette recherche correspond aux deux protocoles présentés précédemment, UPnP et DLNA. Les services sont principalement des services de contrôle et gestion des équipements du domicile, du multimédia au chauffage.

Le service peut aussi être déployé entre deux domiciles (2), chaque HN instanciant une partie du service. Il s'agit par exemple de partage de contenus multimédia ou encore de communication entre domiciles, voire de jeux. Un utilisateur contacte d'autres utilisateurs pour accéder directement aux services disponibles à leur domicile. Cela suppose que les utilisateurs connaissent déjà leurs adresses de contact respectives et qu'il n'y a pas de processus de recherche.

Les services inter-home peuvent vite être augmentés pour passer à l'échelle d'une communauté (3) voire à celle du multi-home (4), à savoir entre tous les HNs d'un même opérateur. Le service communautaire consiste principalement à partager des ressources au sein du réseau social constitué par une communauté d'utilisateurs. Si

celle-ci s'étend à tous les domiciles de l'opérateur de service, on arrive à la solution 4. Les frontières entre les localisations 2, 3 et 4 sont étroites puisqu'il peut s'agir parfois d'une simple question de point de vue. En effet, doit-on considérer que si un service est présent sur toutes les boxes, il s'agit d'un service de type 4 ? Si le service se sert de tous les domiciles, c'est bien le cas. Par exemple un service statistique qui recenserait l'avis sur un programme télévisé serait soit communautaire s'il se restreignait à vos amis soit multi-home s'il vérifiait l'avis de chacun des domiciles. Toutefois, il pourrait avoir un fonctionnement proche de l'inter-home, s'il allait chercher cette information chez votre voisin qui a fait la requête avant vous.

En plus de l'interaction entre les HNs, l'opérateur peut aussi avoir un rôle, en proposant une partie du service dans son réseau (5). C'est classiquement le cas pour le *triple play*. Il est aussi envisageable de gérer l'indexation des ressources, l'authentification ou encore d'autres aspects de gestion et de facturation.

Enfin tout ou partie du service peut être en dehors du réseau de l'opérateur en utilisant l'accès Internet pour par exemple se connecter à un serveur (6).

1.5.2.2. Accès au service

En ce qui concerne l'accès au service par l'utilisateur, il y a trois possibilités comme présentés précédemment dans la section 1.4.1 et la Figure 1.4.1-1.

Dans le cadre d'un accès direct au HN (utilisateur à l'intérieur du domicile), il peut être préférable de rajouter des mécanismes d'authentification et de gestion de droits d'accès. L'utilisateur peut être le propriétaire des services, ou un utilisateur « invité ». La HGw peut prendre en charge l'authentification des utilisateurs et leurs autorisations en fonction de leur profil utilisateur. Cette fonctionnalité doit pouvoir être configurée par le propriétaire des services (par exemple offrir l'accès wifi pour l'invité et l'accès aux fichiers audio mais pas aux photos et la gestion du climatiseur)

Lors d'un accès distant, les problèmes d'authentification et d'autorisation se pose à nouveau, avec en prime la nécessité d'une communication sécurisée. C'est encore la HGw qui est le point de contrôle pour accéder au HN. Toutefois, on peut imaginer des solutions où l'opérateur offre ce service de manière centralisée.

Dans le cadre de communications entre domiciles se pose la question de l'interaction entre les deux HNs. Le HN devant rendre un service à l'autre HN doit être en mesure de contrôler l'identité de son correspondant et de connaître quels sont ces droits d'accès aux ressources et services. C'est dans ce cadre que peuvent intervenir la gestion de communauté ainsi que des profils plus complexes.

1.5.3. Type de partage

Cet aspect est lié à la nature de l'objet partagé entre les utilisateurs de service. La ressource peut prendre plusieurs formes : le contrôle d'équipements, des données multimédia, ou encore un service lui-même.

Dans la section traitant des HS, nous avons déjà abordé les services de contrôle d'équipement, qui sont la base des services d'automatisation du domicile, mais aussi du *Home Theater* qui peuvent être facilement mis en place par DLNA ou UPnP.

Un autre type de partage possible est celui des données multimédia. Une fois que l'utilisateur connaît l'adresse de la ressource qu'il souhaite acquérir, il peut directement la télécharger en se connectant au domicile d'un utilisateur qui possède la ressource et qui accepte de la partager.

Enfin le partage de services lui-même est la dernière opportunité considérée dans ce travail. Ce type de partage peut permettre à l'utilisateur de déployer de nouveaux services. De même que pour un contenu, l'utilisateur cherche un service le récupère et déploie son service à domicile. Celui-ci peut avoir été certifié par une autorité (comme l'opérateur réseau). Il peut être envisagé des étapes pour vérifier les spécifications matérielles, les droits d'accès, droit de copie, la présence de virus etc… Dans cette catégorie, on peut trouver un grand nombre d'utilisation, cela peut être un service pour commander automatiquement du fuel en fonction de la réserve dans la cuve, comme un service de streaming photo téléchargé chez un ami pour visualiser son album photo pour une durée donnée sans pouvoir les sauvegarder.

1.6. Architectures d'interconnexion de domiciles

Devant la diversité des services à domicile, les Home Services (HS) nécessitent différents besoins en termes de communications, que nous avons classifiés dans la partie précédente. Si les services internes au domicile nécessitent un moyen de communication locale qui peut être réglé par UPnP ou DLNA, la mise en œuvre de HS inter-domiciles ou communautaires requiert d'autres moyens de communication. L'enjeu de notre travail est alors de proposer des architectures permettant une mise en œuvre sécurisée des HS inter-domiciles ou communautaires. Ces solutions sont gérées tout ou en partie par l'opérateur du réseau.

1.6.1. Architectures centralisées

Au-delà de l'utilisation classique d'Internet qui consiste à un mode client/serveur, il existe de nombreuses solutions héritées de ce modèle. Cette partie propose d'aborder l'interconnexion via des solutions centralisées.

1.6.1.1. Virtual Private Network (VPN)

Une première solution assez classique chez les opérateurs est le VPN. L'accès aux services autre qu'à domicile passe par un serveur de VPN installé dans le cœur du réseau. Via ce dernier, un client d'un HN peut s'authentifier et accéder à d'autres HNs. Les principales technologies sont IPSec, SSL, MPLS et L2TP.

Il y a beaucoup de scénarios d'interconnexion de cette technologie, puisqu'elle permet aussi bien à un utilisateur distant d'accéder à son HN comme à deux HNs de s'interconnecter. Toute solution vise en effet à créer une connexion avec un serveur

VPN qui est en mesure d'accéder à l'HGw (Home Gateway) décrit comme dans la Figure 1.6.1-1.

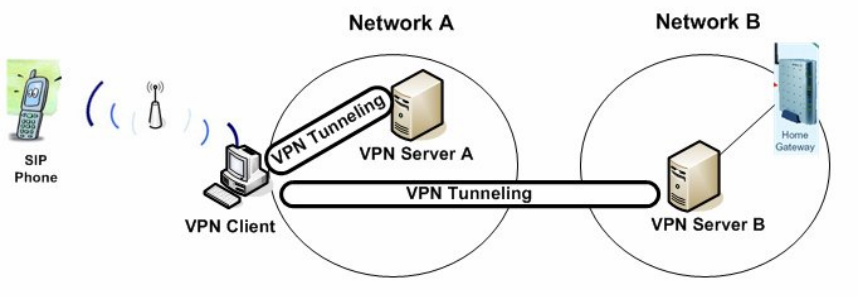

Figure 1.6.1-1: Interconnexion via VPN

Dans le cadre de ce travail, la solution VPN n'a pas été plus avant étudiée dans la mesure où ce type de méthodes a été largement étudié et utilisé par les opérateurs. Il s'agit d'ailleurs d'une solution très fréquente dans l'accès Internet lui-même avec des protocoles tels que L2TP.

1.6.1.2. Session Initiation Protocol (SIP)

SIP [RSCJ02] est à la fois un protocole qui permet le contrôle de session applicative (ouverture, fermeture et modification) et une architecture de communication avec différentes fonctionnalités (proxy, registre, serveur, …). Cette architecture de communication repose sur un modèle centralisé en ce qui concerne la signalisation.

Principe

A l'origine, le but de SIP était de gérer les invitations de nouveaux clients à une téléconférence déjà existante. Désormais, ses fonctionnalités se sont largement diversifiées, gérant une session multimédia dans son intégralité et proposant un certain nombre d'extensions (messagerie instantanée, notifications d'événements, etc.). Toutefois si l'on reste dans le cadre de l'IETF, SIP a une mission bien définie : délivrer la description d'une session multimédia aux utilisateurs, et ce en fonction de leur emplacement actuel. Cette description peut utiliser le *Session Description Protocol* (SDP) [HaJa98] ou un tout autre format puisque SIP est indépendant du format de description des sessions. Dans la pratique SDP est utilisé dans la quasi-totalité des produits. Pour le transport des données multimédias, c'est RTP [SCFJ03] qui est utilisé. La figure suivante (Figure 1.6.1-2) illustre une transaction SIP. Ici SIP gére la session et une fois celle-ci établie, le ou les flux multimédia sont véhiculés sur un chemin qui peut être différent via RTP.

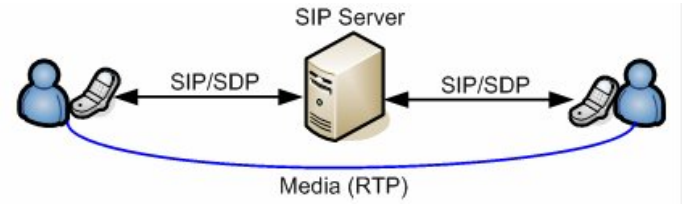

Figure 1.6.1-2: Signalisation du SIP d'appel

Identification et localisation
Un client SIP est identifié par un ou plusieurs *Uniform Resource Identifiers* (URIs) en suivant le RFC 2396 [LeFi98]. Les SIP URIs sont largement inspirés d'une adresse mail mais, contrairement à elle, un URI peut contenir des paramètres.

Pour pouvoir identifier un client, celui-ci doit avoir un seul et unique identifiant, et peu importe sa localisation. Cet identifiant est appelé l'URI public ou encore son adresse d'enregistrement (*Address of Record* AoR). L'utilisateur peut toutefois utiliser plusieurs terminaux de différente nature (smartphone, PC, télévision, etc.). Pour différencier les terminaux d'un même client, SIP utilisent différents URIs, que l'on appellera ici les URIs courants.

Pour faire le lien entre l'URI public d'un client SIP et son URI courant, une correspondance est mise en place en utilisant un système d'enregistrement auprès d'une entité spécifique. A chaque nouvelle session ouverte par un utilisateur, l'enregistrement de son URI courant a lieu auprès de son registre (*registrar*). Ainsi toute requête à son AoR est envoyée au registre de son domaine et ce dernier prend en charge de la faire suivre puisqu'il connaît sa localisation réelle.

Principales entités SIP
L'architecture SIP se fonde sur quatre entités:
- les agents (*user agents* - UA) sont les extrémités de l'architecture SIP. Il existe une grande variété d'UA : téléphone portable, téléphone SIP, logiciel sur ordinateur, PDAs, etc. ... Un UA peut aussi bien être l'équipement terminal d'un client qu'un serveur applicatif.
- les registres (*registrars*) sont en charge de la correspondance entre l'adresse publique d'un utilisateur et son adresse courante. Chaque domaine est associé à un registre précis, un registre pouvant gérer plusieurs domaines
- les serveurs de redirection (*redirect server*) sont utilisés pour le routage des messages SIP : lorsqu'une AoR n'est pas atteignable, c'est vers un serveur de redirection que le message SIP est dirigé. En réponse, le serveur envoie une adresse alternative (s'il en connaît une).
- les passerelles (*proxy servers*) sont des entités de routage pour SIP ; une passerelle reçoit donc un message SIP et le relaie au prochain nœud SIP concerné (sa destination ou une autre passerelle).

Format des messages
Les figures présentées dans ces deux dernières parties (Figure 1.6.3-2 et Figure 1.6.3-3) ont esquissé certains mécanismes de SIP. Toutefois, avant d'étudier les différentes transactions, présentons le format des messages échangés.

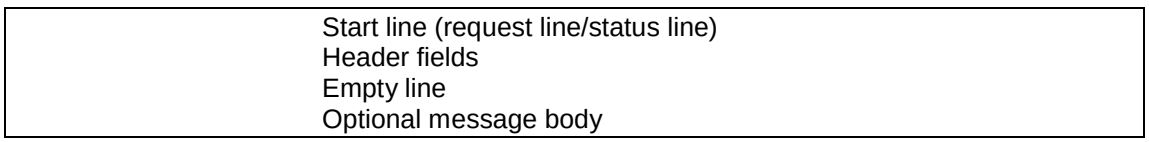

Figure 1.6.1-3: Format d'un message SIP

Comme le montre la Figure 1.6.1-3, un message SIP est constitué d'une ligne de démarrage, d'un certain nombre de champs d'en-tête, d'une ligne vide et d'un corps de message qui peut-être optionnel (c'est dans cette partie que l'on trouve notamment la description SDP d'une session en cours de négociation).

Les messages SIP se divisent en deux types différentiés par la *start line*: les demandes (*request line*) et les réponses (*status line*).

Une requête (*request line*) présente le nom de la méthode, suivi de l'URI de la destination et de la version du protocole [RSCJ02]. Il existe plusieurs méthodes SIP que l'on retrouve dans le Table 1.6.1-1[1].

Table 1.6.1-1: Liste des différentes méthodes SIP

Méthode	***Utilisation***
ACK	acquittement de la réponse définitive à un message INVITE
BYE	fin d'une session
CANCEL	annulation d'une requête en attente
INFO	message contenant de la signalisation téléphonique (PSTN)
INVITE	demande d'ouverture de session
NOTIFY	notification d'événement à un client SIP
OPTIONS	demande d'informations sur les capacités d'un serveur
PRACK	acquittement d'une réponse provisoire à un message INVITE
PUBLISH	exportation d'informations vers un serveur
REGISTER	enregistrement de la correspondance entre AoR et URI courante
SUBSCRIBE	demande d'enregistrement à un service de notification
UPDATE	modification des caractéristiques d'une session en cours
MESSAGE	message instantané
REFER	demande à un serveur d'envoi d'une requête

Une ligne de statut (*status line*) est constituée d'un numéro de version et du statut de la transaction sous la forme d'un code et d'une phrase[2]. Les codes des différents statuts sont compris entre 100 et 699, classés par centaine (Table 1.6.1-2).

Table 1.6.1-2: Liste des différents types de codes retours

Codes	***Types***
De 100 à 199	Information
De 200 à 299	Succès

[1] Certaines méthodes présentées dans ce tableau ont été introduites dans le cadre d'extensions de SIP, comme PRACK.
[2] Cette phrase n'a qu'une valeur de lisibilité.

Codes	*Types*
De 300 à 399	Redirection
De 400 à 499	Erreur – niveau client
De 500 à 699	Erreur – niveau serveur
De 600 à 699	Erreur globale

Mis à part cette première ligne, requêtes et réponses sont constituées de la même manière. Dans l'exemple suivant, nous illustrons une requête SIP qui est envoyée à Alice par Bob. Bob utilise la méthode INVITE pour demander un nouvel appel. L'en-tête *via* piste le chemin de l'appelant vers l'appelé. Ici il s'agit de l'UA de Bob. Le champ *call-ID* est un identifiant unique pour suivre cet appel particulier, il a été généré par l'UA appelant. *CSeq* est utilisé pour compter les messages dans un appel et ainsi appairer requêtes et réponses (Figure 1.6.1-4).

```
INVITE sip:bob@enseeiht.fr SIP/2.0
Via: SIP/2.0/UDP yon.enseeiht.fr;branch=z9hG4bK776asdhds
Max-Forwards: 70
To: Alice <sip:alice@irit.fr>
From: Bob <sip:bob@enseeiht.fr>;tag=1928301774
Call-ID: a84b4c76e66710@yon.enseeiht.fr
CSeq: 314159 INVITE
Contact: <sip:bob@enseeiht.fr>
Content-Type: application/sdp
Content-Length: 142
(Bob's SDP not shown)
```

Figure 1.6.1-4: Exemple de requête SIP d'invitation

Dialogue SIP

La première étape avant de pouvoir communiquer avec SIP est de s'enregistrer auprès du registre de son domaine. Cette étape, si elle n'est pas couplée à une authentification, se déroule en une simple transaction REGISTER/OK.

Le terme dialogue SIP est employé pour l'échange de messages entre deux UAs pour établir, modifier et terminer une session. Un dialogue classique se caractérise par la création d'une session (transaction INVITE-ACK), suivie de l'échange de données puis de la terminaison de cette session (Figure 1.6.1-5).

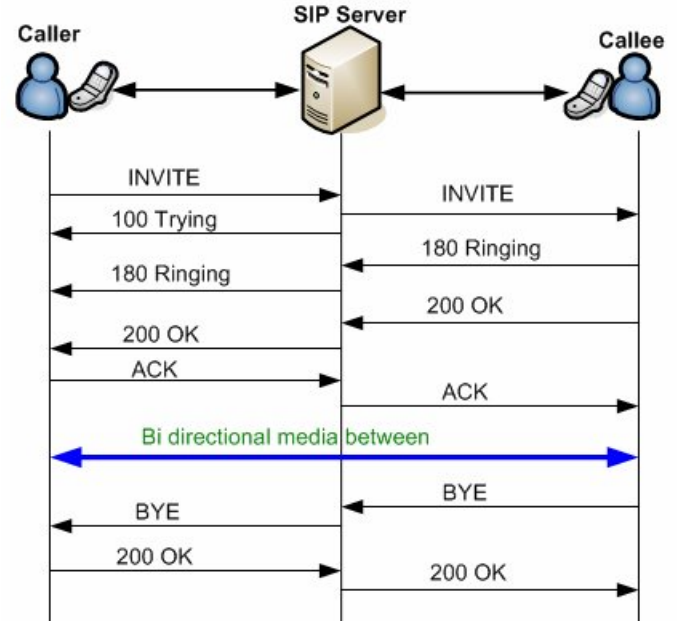

Figure 1.6.1-5: Illustration d'une session SIP classique

1.6.1.3. IP Multimedia Subsystem (IMS)

IP Multimedia Subsystem (IMS) est une architecture *Next Generation Network* (NGN) standardisée qui permet de fournir des services multimédias fixes et mobiles. C'est la solution de NGN du monde de la téléphonie mobile. Les principaux objectifs d'IMS sont l'intégration de services et l'intégration des technologies d'accès.

Concernant l'intégration des réseaux d'accès, la version d'IMS standardisée par le 3GPP [3GPP12] est plus axée sur les technologies du monde du mobile. Toutefois, d'autres technologies de réseau sans fil et filaires tels que le Wimax, Wifi, xDSL, câble à large bande ont été également intégrées dans IMS, notamment par le travail du groupe TISPAN (*Telecommunication and Internet Converged Services and Protocols for Advanced Networking in European Telecommunications Standards Institute*) [ETSI11].

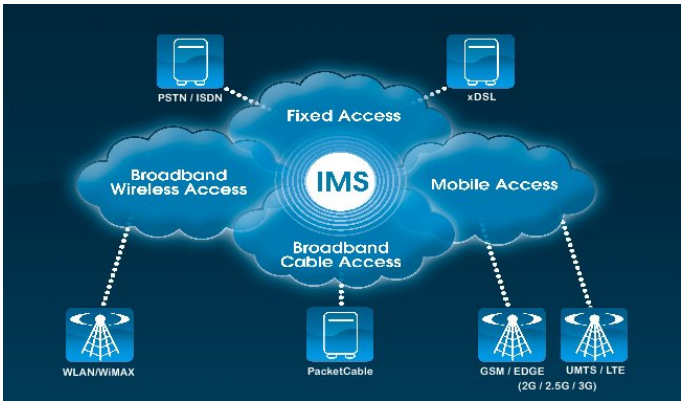

Figure 1.6.1-6: Vue simplifiée d'IMS par TISPAN [ETSI11]

La définition d'IMS est très dépendante de l'entité qui le considère. Pour l'ETSI, via notamment TISPAN, il s'agit d'offrir une convergence entre les technologies d'accès comme la Figure 1.6.1-6 l'illustre. Sur cette figure apparaît clairement l'objectif principal de l'architecture IMS telle qu'envisagée par TISPAN : l'intégration de différents supports physiques. On retrouve ici les différents média physiques du 3GPP (2G, 2.5G et 3G) mais aussi les réseaux d'accès filaires de type xDSL, RTC, RNIS et câble. Enfin les accès sans-fil sont aussi intégrés avec les WLAN et le WiMAX. IMS est alors considérée ici comme une architecture de convergence des différents réseaux d'accès, sans fil et filaire, avec les réseaux de mobiles.

Pour un constructeur comme CISCO, la présentation IMS est différente (Figure 1.6.1-7). Ainsi, on peut observer un autre point clef de l'architecture IMS ; au-delà de la convergence des supports, c'est l'intégration des différents services qui est visée en offrant une plateforme commune, facilitant leur déploiement et centralisant des fonctionnalités comme la gestion de session, les fonctions d'authentification, autorisation, et traçabilité connues sous le sigle AAA (Authentication, Authorization, Accounting/Auditing) ou encore la gestion de la QoS.

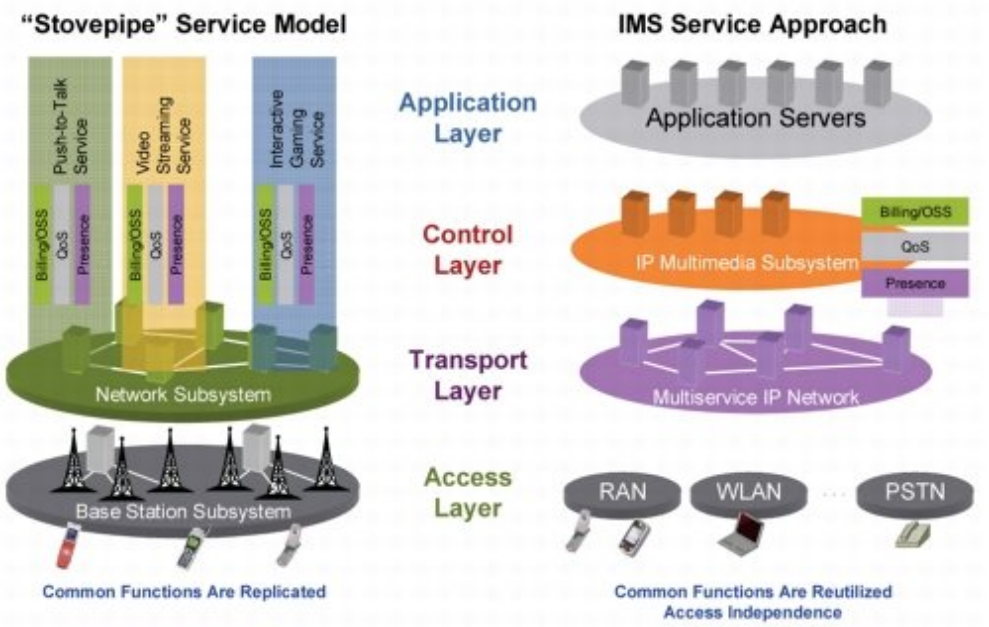

Figure 1.6.1-7: Vue simplifiée d'IMS par CISCO

Architecture IMS

L'architecture IMS propose une vision en trois niveaux (Figure 1.6.1-8) :

- la couche réseau (parfois appelé transport de données) est le support à la communication. Chez un opérateur mobile il s'agit du réseau LTE ou UMTS.
- la couche de contrôle de session est le cœur d'IMS. Ce niveau s'occupe du AAA et de la gestion des communications. C'est le protocole SIP qui a été choisi par le 3GPP pour réaliser cette fonction.
- la couche service qui est généralement indépendante d'IMS. Il s'agit de serveurs applicatifs, principalement SIP.

On peut noter la présence sur la figure d'un quatrième niveau, il s'agit d'une couche d'adaptation entre le monde paquet et le monde circuit via les Media Control Gateways. Toutefois, il faut noter que comme la plupart des standards ITU, IMS est plus adaptée à une description fonctionnalités/interfaces qu'un modèle en couche.

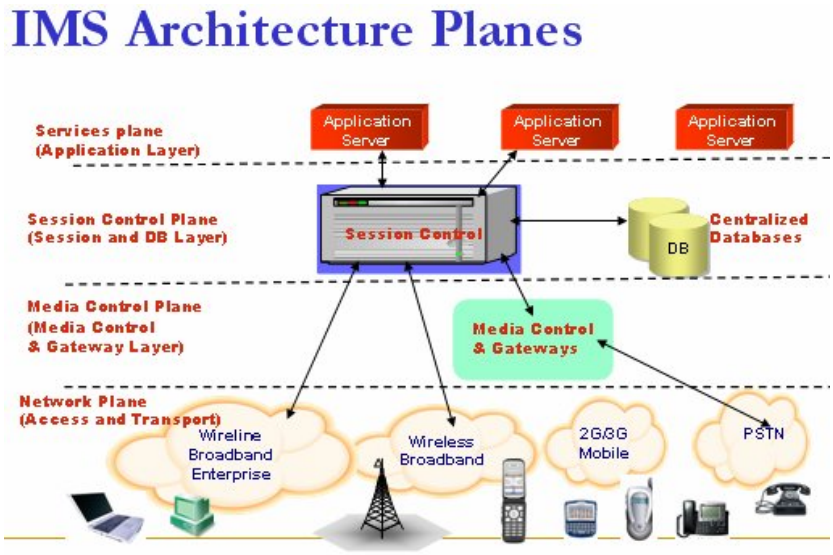

Figure 1.6.1-8: Une vision en « couches » d'IMS [Zeb06]

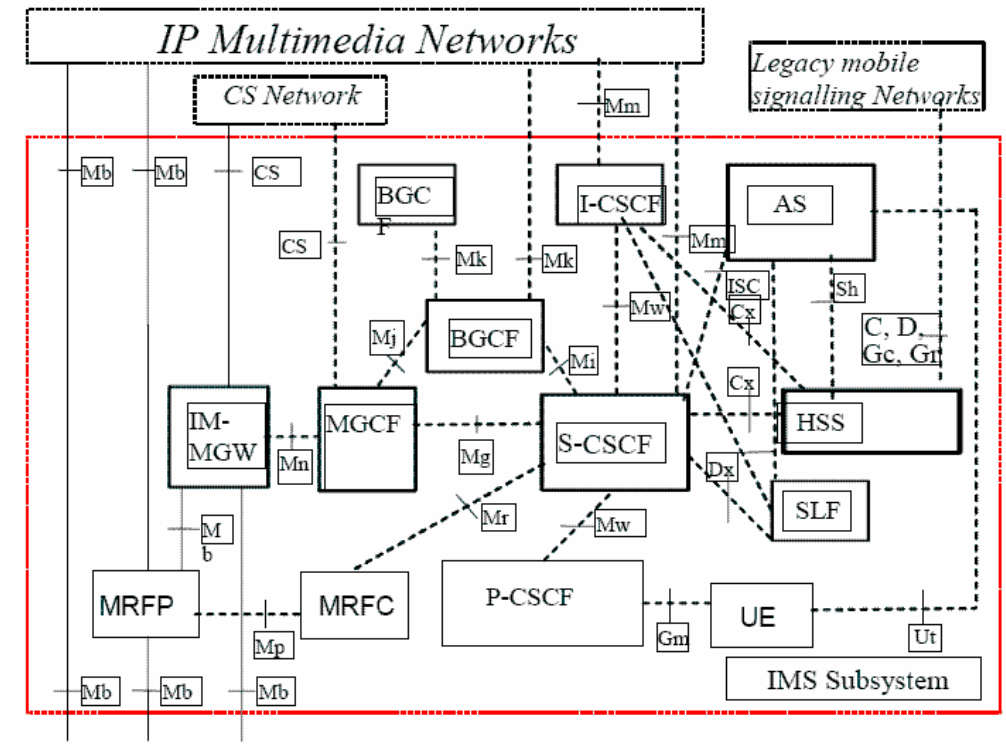

Figure 1.6.1-9: Architecture de référence du sous-système de cœur IMS [3GPP12]

Principales entités de l'architecture IMS

Le modèle d'architecture IMS définit des fonctions et des interfaces dont la totalité est représentée par la Figure 1.6.1-9 issue du standard [3GPP12]. Précisons à la suite les principales entités (ou fonctionnalités) de cette architecture.

Le Home Subscriber Server (HSS) est une entité qui stocke et gère les informations de chaque client de l'opérateur : identité, nom et localisation du serveur S-CSCF associé, profil d'itinérance, informations d'authentification ou encore privilèges de service. Les entités de contrôle de session, à savoir l'I-CSCF et le S-CSCF, ont besoin d'accéder aux éléments de cette base de données pour trouver et gérer un client IMS. C'est le protocole *Diameter* [GBPC06] une évolution de *RADIUS* [RWRS00], qui est utilisé pour ce dialogue.

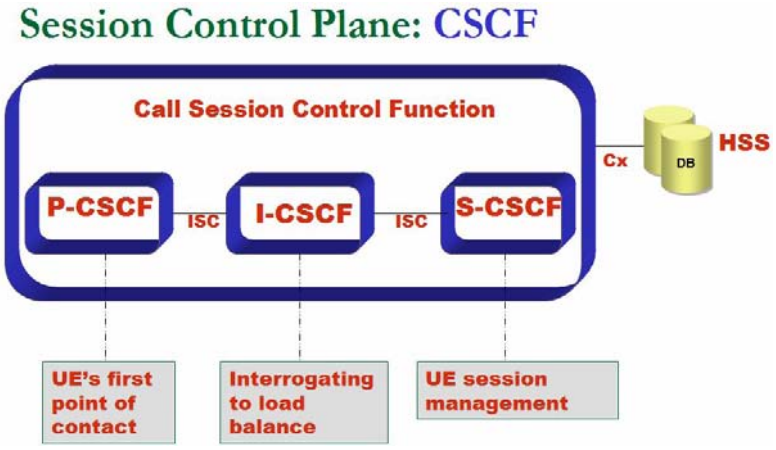

Figure 1.6.1-10: Les trois entités de contrôle de session [Zeb06]

Les Call Session Control Function (CSCF) sont les éléments de cœur d'IMS ; ils prennent en charge les différents contrôles de session des clients IMS en traitant et générant la signalisation SIP. Ils jouent souvent un rôle dans l'allocation de ressources. Il existe trois types de CSCF (Figure 1.6.1-10) :

- P-CSCF (Proxy-CSCF) : C'est le premier point de contact de l'utilisateur à IMS et il a le rôle d'une passerelle. Il traite toute signalisation provenant ou destinée aux utilisateurs dont il a la charge, servant les requêtes en interne ou les relayant. Il peut être situé soit dans le réseau visité, soit dans le réseau domestique. Afin d'accéder à ses services IMS, un terminal utilisateur doit découvrir un P-CSCF via une procédure qui peut être indépendante de la technologie d'accès, comme DHCPv6/DNS, ou non, comme dans le *General Packet Radio Service* (GPRS) où la procédure est la récupération du *Packet Data Protocol* (PDP) *Context*.
- I-CSCF (Interrogating CSCF) : situé en bordure d'un réseau opérateur, c'est à la fois le point de contact pour toutes les connexions destinées à un utilisateur qui s'y trouve, fut-il itinérant (roaming user), et le point d'accès à ce réseau. Il a notamment en charge la localisation d'une entité IMS, la localisation (ou l'affectation) d'un S-CSCF en charge d'un client IMS appartenant à ce réseau opérateur et le routage des requêtes vers la destination appropriée.
- S-CSCF (Server-CSCF) : cette fonction enregistre les utilisateurs et leur fournit l'accès à leurs services. Le serveur s'occupe de l'authentification de ses clients,

des autorisations via le profil utilisateur et de la mise en place des services demandés par l'utilisateur, notamment en terme de QoS. Le S-CSCF a de nombreuses fonctionnalités telles que :

- la gestion des enregistrements SIP des utilisateurs, liant l'emplacement courant de l'utilisateur (URI privé) et à son identité (URI public).
- le contrôle de toute signalisation SIP provenant et à destination de son client.
- la sélection des serveurs d'applications.

Les serveurs d'application (AS) fournissent des services de tous types aux clients abonnés. Ils ont des interfaces avec le S-CSCF qui utilisent SIP (Figure 1.6.1-11). Selon le service, l'AS fonctionne selon différents modes : s'il ajoute des informations au cours d'une conversation il peut s'agir d'un simple proxy (voix). S'il est le destinataire il a un rôle d'agent utilisateur (messagerie vocale) ; de même en mode Back-to-Back, il joue le rôle d'agent pour plusieurs clients (vidéoconférence). Un AS peut être situé dans le réseau domestique, dans la partie service du réseau d'opérateur ou détenu par une entité tierce. Pour terminer, il faut noter que même si les serveurs SIP sont privilégiés par IMS, on peut interfacer d'autres AS à IMS via des passerelles. Ainsi les services OSA (Open Service Access), communs dans le monde du mobile, sont interfacés à IMS via la passerelle OSA-SCS (OSA Service Capability Server). De même les services CAMEL (Customised Applications for Mobile networks Enhanced Logic) peuvent être utilisés sur IMS via la passerelle IM-SSF (IP Multimedia - Service Switching Function).

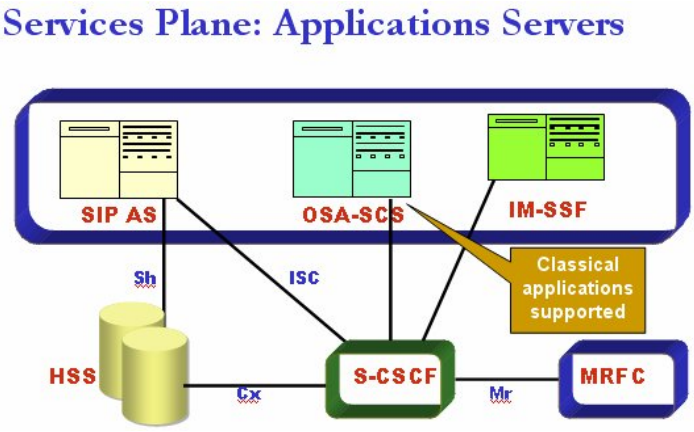

Figure 1.6.1-11: Les trois types de serveurs applicatifs [Zeb06]

Il existe bien d'autres entités IMS. La plupart comme la Signalling Gateway Function ou les Breakout Gateways ont des rôles de passerelles qui raccordent IMS au plus grand nombre de réseaux possibles. Nous n'irons pas plus loin sur ces points dans cette étude, les aspects sessions et services étant les éléments clefs du travail que nous avons mené.

Exemple d'appel d'IMS

Cet exemple (Figure 1.6.1-12) illustre comment des appels multimédias peuvent être coordonnées par IMS. Il s'agit ici d'une session multimédia entre deux clients IMS de

deux réseaux différents. Par convenance, nous les appellerons Alice et Bob. L'appel est initié par Alice.

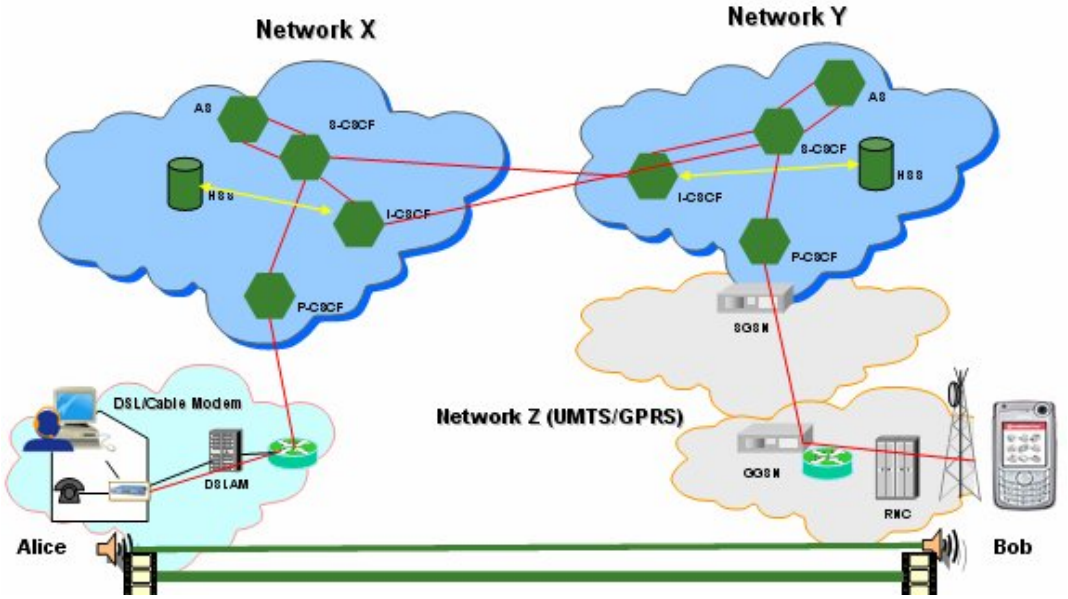

Figure 1.6.1-12: Exemple d'un appel entre deux réseaux IMS [Mili06]

1. L'appel vocal vient d'Alice et est tout d'abord traité par son réseau IMS, X. C'est le P-CSCF qui reçoit la signalisation d'Alice. Il extrait la demande d'Alice et vérifie son origine et sa conformité.
2. Le P-CSCF transmet l'appel au S-CSCF d'Alice, situé dans le réseau mère X
3. Le S-CSCF traite sa requête, exécute le contrôle du service qui peut inclure des interactions avec le serveur d'applications et détermine le point d'entrée de l'opérateur de la maison d'Alice en se basant sur l'identité d'Alice incluse dans la demande SIP INVITE.
4. Le S-CSCF transmet l'appel à l'I-CSCF du réseau Y.
5. L'I-CSCF interroge le HSS du réseau Y pour déterminer le S-CSCF et lui transmet la demande d'appel.
6. Le S-CSCF du réseau Y prend en charge le traitement de la session, interroge le serveur d'applications pour les services de terminaison.
7. Le S-CSCF transmet l'appel au P-CSCF du réseau Y assigné pour Bob via le réseau Z (UMTS / GPRS) et une négociation peut être entamée entre les deux UE pour ouvrir la communication.

Le détail de cet échange est donné dans la Figure 1.6.1-13. Trois étapes principales se degagent : la demande d'ouverture d'appel, la négociation des ressources (des codecs utilisés à la demande de ressource sur les réseaux sous-jacents) et l'acceptation de l'appel. Il est à noter que ces échanges peuvent induire un coût de signalisation non négligeable.

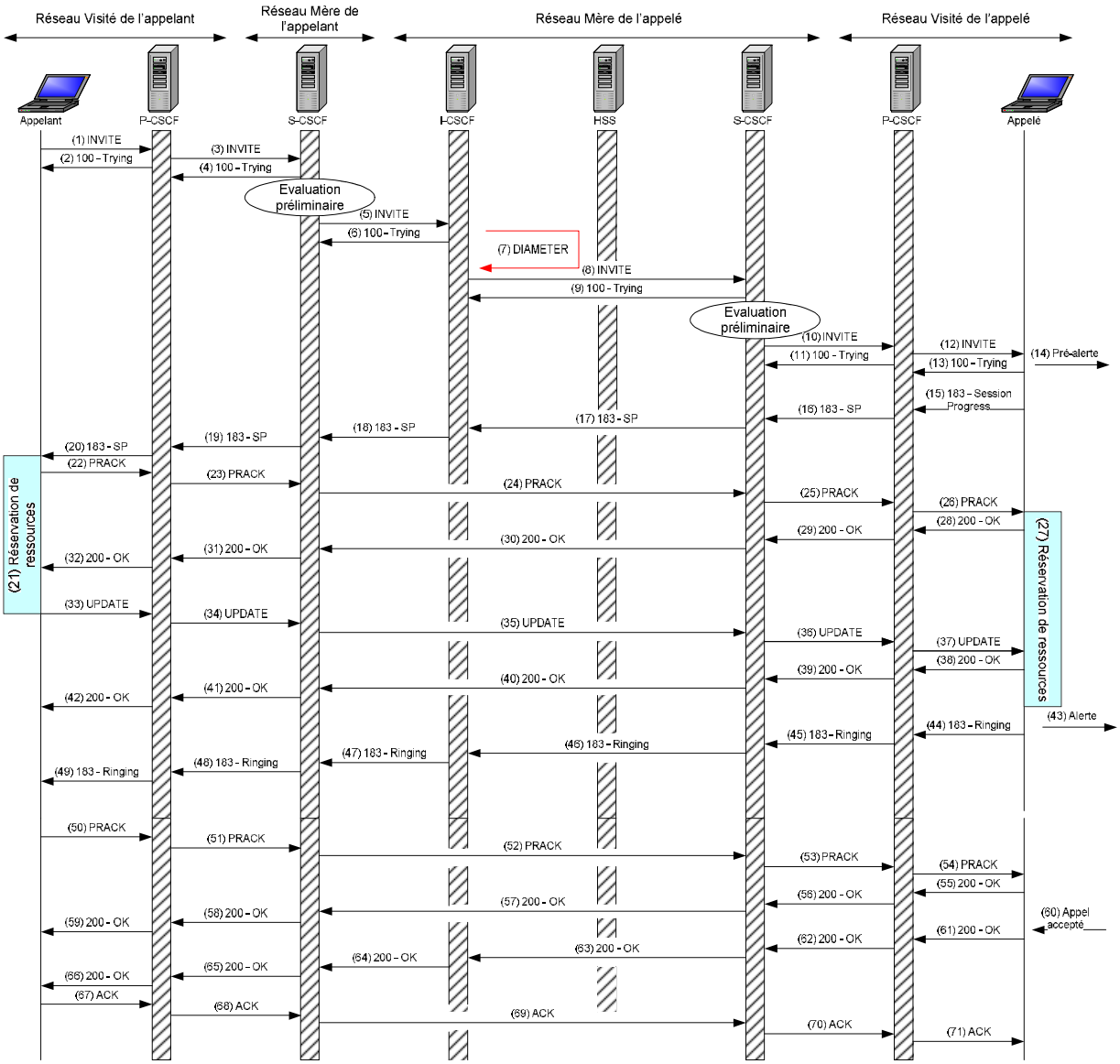

Figure 1.6.1-13: Ouverture d'une session IMS entre deux clients IMS de réseaux différents

1.6.2. Architectures décentralisées (Pair-à-Pair)

Face aux modèles centralisés, d'autres types d'architectures ont vu le jour, plus proches du concept de mise en partage d'Internet. Ces architectures sont communément dénommées pair-à-pair et c'est l'objectif de cette section de les présenter pour pouvoir ensuite les utiliser dans l'interconnexion de HN. Leur intérêt est de faire face aux problèmes du modèle client serveur tels la défaillance du serveur ou encore le risque de goulot d'étranglement.

1.6.2.1. Définition de P2P

Le terme P2P a été médiatiquement très utilisé depuis 2005, mais il en est pas moins un concept assez difficile à définir. En se reférant aux utilisations très nombreuses du pair à pair deux notions différentes se font jour : celle d'applications P2P et celle de réseau P2P.

Les services P2P
Les services P2P sont très majoritairement des services de partage et téléchargements de données, mais aussi des services de publications, de travail collaboratif, de dépôt, de voix sur IP, de calcul sur le génome… Classifier le P2P par ses applications est alors une tâche très difficile puisque la plupart des applications peuvent être des applications P2P. Une solution est alors de statuer sur le fait qu'une application P2P est une application qui utilise un réseau P2P.

Les réseaux P2P
L'architecture P2P trouve un grand nombre de définitions, mais il est presque toujours possible de trouver un contre-exemple à ces définitions.

Il y a la définition historique avec Napster (mai 1999) qui est considéré comme le premier système informatique P2P. Le système est fondé sur un serveur central pour indexer l'information (ici des fichiers audio au format MP3), mais pas télécharger ni fournir des données. Pour éviter une trop grande utilisation des ressources du serveur les fichiers une fois localisés sont téléchargés directement sur les détenteurs. Ainsi l'information se dissémine et les « téléchargeurs » d'hier deviennent les sources d'aujourd'hui. Dans ce système tous les éléments n'ont pas exactement le même rôle, ils ne sont pas réeellement pairs et les puristes ne le considèrent pas comme du P2P.

Au-delà de la notion de pair, la littérature propose plusieurs définitions comme :

«Une architecture de réseau distribué peut être appelée un réseau P2P, si les participants partagent une partie de leurs ressources matérielles propres (par exemple, la capacité de traitement, capacité de stockage, la capacité de liaison du réseau, imprimantes). Ces ressources partagées sont nécessaires pour fournir le service et le contenu offert par le réseau (partage de fichiers par exemple ou des espaces de travail partagés pour la collaboration), de sorte qu'elles deviennent accessibles par d'autres pairs» [Scho01].

«Les réseaux P2P sont des systèmes distribués composée de nœuds interconnectés capables de s'organiser en topologie de réseau dans le but de partager des ressources telles que le contenu, les cycles de CPU, de stockage et de bande passante. Ils sont capables de s'adapter aux erreurs, pertes et à la forte volatilité des nœuds tout en maintenant la connectivité et des performances acceptables, et cela sans avoir recours à un serveur global centralisé ou d'autorité» [ThSp04].

Malheureusement ces définitions excluent des solutions de P2P plus centralisées. Pour simplifier, une façon de définir une architecture réseau de P2P est alors de qualifier

par défaut toute communication non purement client/serveur de pair-à-pair. Dans les faits, il y a deux extrémités : le modèle pur client/serveur et le modèle pur P2P. Au milieu, on trouve des variations plus ou moins proches de l'un ou de l'autre [Cama09]. Si dans les communications, il y a une communication directe entre les clients et si, dans une certaine mesure, tout nœud rend un service à l'ensemble et peut être servi par cet ensemble, on pourra parler de P2P.

1.6.2.2. Types d'architectures de P2P

Une question de taxonomie

La difficulté de la définition du P2P va de pair avec la profusion de critères de classification du P2P. Il est ainsi possible de classer les architectures en fonction de leur génération et donc de leur date d'apparition. Toutefois, cette classification mélange de nombreux critères car, après NAPSTER, le P2P a évolué sur plusieurs voies parallèles. Une autre solution est de classer les architectures en fonction de la ressource qu'elles partagent : types de données, services, bande passante, ressource CPU. On trouve aussi des classifications reposant sur le degré d'anonymat ou d'autres types de sécurité que l'architecture peut fournir.

Pour cette étude nous suivrons une classification plus conventionnelle, celle du degré de centralisation et de la structuration de l'architecture, à savoir comment données sont indexées par le système. L'indexation peut-être centralisée, semi-centralisée, décentralisée sans structure ou encore décentralisée structurée. Pour ce dernier cas on parle souvent de P2P structuré en opposition avec le P2P pur sans organisation d'indexation.

Le P2P centralisé

Pour ce type de P2P, il existe une entité centrale qui indexe les ressources partagées (Figure 1.6.2-1), et gère parfois les profils utilisateurs. Ce type d'architecture est la plus à même de mettre en œuvre des protocoles AAA. Toute nouvelle ressource doit être enregistrée auprès du serveur central et toute requête passe par cet index pour trouver le ou les emplacements où celle-ci est disponible.

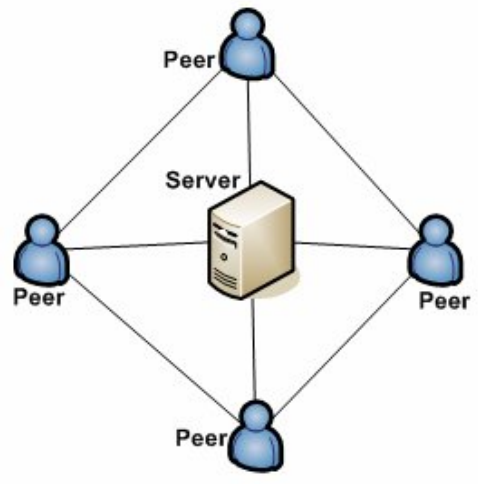

Figure 1.6.2-1: Illustration d'une architecture P2P centralisée

La construction de la topologie est très simple : rejoindre l'architecture revient à se connecter au serveur central. Comme exemple de ces architectures, on peut noter NAPSTER, Audiogalaxy ou dans une certaine mesure BitTorrent (même si la notion de ressource indexée est différente). Ce type d'architecture a été repris notamment par certaines technologies de voix sur IP ou encore les architectures des sites communautaires.

Le P2P pur (P2P décentralisé sans infrastructure)
Dans un réseau P2P pur, chaque nœud est responsable de ses ressources et l'indexe directement, sans serveur central ou aucune autre entité. L'indexation est donc une étape directe qui ne nécessite aucune signalisation. En revanche, trouver la ressource revient à trouver le nœud qui la détient. Pour cela, des messages inondent le réseau. Cette signalisation peut être très coûteuse, surtout quand la ressource est rare sur le réseau.

La topologie elle-même est construite de voisin en voisin (Figure 1.6.2-2). Pour rejoindre le réseau, un nœud doit disposer de l'adresse d'autres nœuds du réseau. Il peut utiliser le cache de ses anciennes connexions ou encore des nœuds de *bootstrap* obtenus par d'autres moyens. Chaque nœud maintient une liste de ses voisins actifs en utilisant des messages de type hello ou ping/pong (Gnutella). Le maintien de l'architecture et la recherche de ressources peuvent être très couteux en termes de signalisation. L'exemple le plus connu de cette architecture est Gnutellav0.4, une initiative d'AOL ou encore Freenet. Le pur P2P est à l'origine de solutions de P2P hybrides ou encore de réseaux de capteurs.

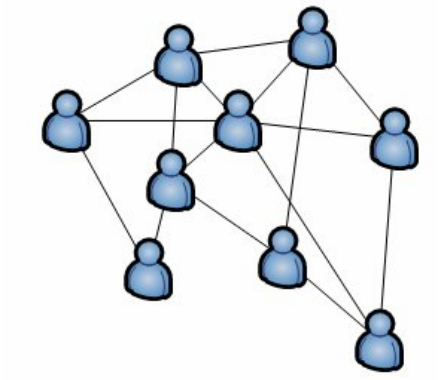

Figure 1.6.2-2: Illustration d'une architecture de pur P2P

Le P2P hybride (P2P semi-centralisé)
Le P2P hybride est une évolution du P2P pur qui réduit la signalisation par une hiérarchisation. Le P2P hybride ajoute un index local, dénommé super-pair. A l'échelle locale, les nœuds fonctionnent comme en mode centralisé avec le super-pair tandis que les super-pairs construisent entre eux un réseau de type pur P2P (Figure 1.6.2-3).

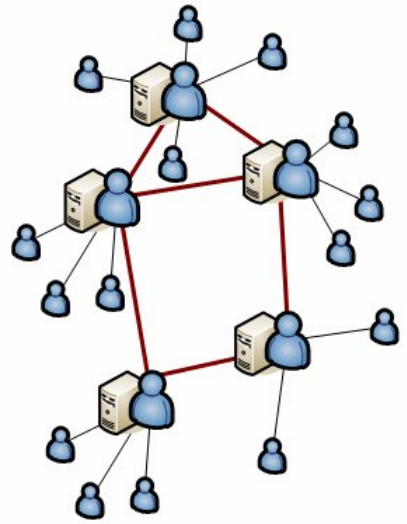

Figure 1.6.2-3: Illustration d'une architecture de P2P hybride

La construction de l'architecture est semblable à celle du pur P2P à la seule exception du choix du super-pair. Ce dernier peut être prédéfini ou bien élu/désigné comme dans Gnutellav0.6. Cette forme de P2P a le plus évolué donnant lieu à de nombreuses variantes telles que Skype, Edonkey2000 …

Le P2P structuré

Pour faire la médiane entre le pur pair-à-pair (recherche en $O(n^2)$) et le pair-à-pair centralisé (état du serveur en $O(n)$), une solution d'indexation distribuée et structurée a été avancée. Elle est fondée sur la notion de Distributed Hash Table (DHT). Le principe est de distribuer l'espace d'adressage des ressources à tous les nœuds constituant le réseau. Pour se faire, les nœuds et les ressources sont adressés dans le même espace à plat. Les ressources et les nœuds obtiennent cette adresse via une fonction de hachage et une métrique impose une relation de distance entre les différents éléments : le nœud le plus proche de la ressource est le nœud qui est en charge de son indexation. On représente souvent l'ensemble de l'espace des ressources/nœuds via un anneau (Figure 1.6.2-4). Ici le nœud N connait un nombre limité de nœuds, ses voisins.

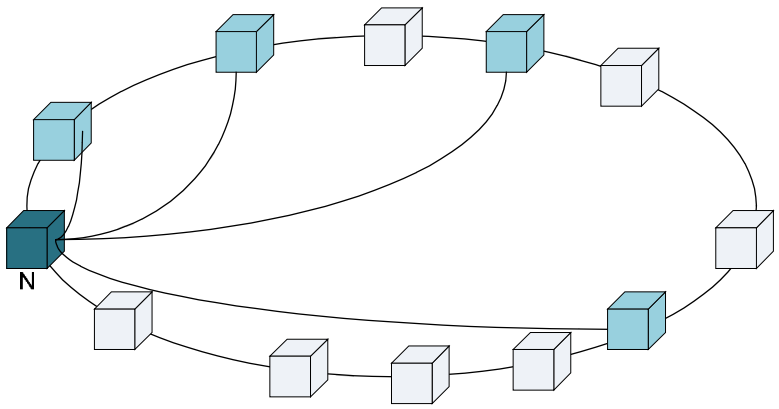

Figure 1.6.2-4: Illustration d'une architecture de P2P structuré

Si cette topologie est efficace en terme de recherche d'information, elle n'en reste pas moins difficile à construire et à maintenir en particulier la gestion des départs et des arrivées s'avère complexe. Face à la volatilité des nœuds, ce type d'architecture n'est pas toujours adapté et est donc plutôt utilisé pour les services de publication (PUBLIUS [WaRC00]) et de base de données avec des nœuds fixes et stables que pour du téléchargement de fichier. De plus il est souvent difficile de rechercher des ressources puisqu'il faut avoir le nom précis de la ressource pour pouvoir la hacher et la trouver.

Le protocole le plus connu de ce type d'architecture est Chord [SMKK01] dont nous étudierons le déploiement dans un chapitre ultérieur. Notons également des solutions comme Can [RFHK01], Pastry [RoDr01], Tapestry [ZhKJ01] ou encore EPICHORD [LeLD04]. Parmi les évolutions de Chord, citons le partage de fichier avec le réseau KAD et son protocole Kademlia [MaMa02]. Ce protocole est assez complexe, puisqu'il essaie de combler les failles de CHORD dans un environnement où 90% des nœuds peuvent disparaître à tout moment. La solution a connu un très grand succès avant de s'avérer très controversée, notamment en termes de performances mais surtout de failles de sécurité [StEB09] aussi nous sommes nous concentrés dans ce travail sur le protocole Chord.

1.6.2.3. Fonctionnement de Chord

Comme précisé ci-dessus, Chord [SMKK01] est un protocole de recherche de ressources qui distribue l'indexation des ressources sur les nœuds du réseau en utilisant des clés (valeur de hachage de la ressource, adresse de la ressource, protocole). Un nœud devient responsable d'une clé si la valeur est comprise entre son identité et l'identité de son successeur.

C'est l'algorithme Chord qui gère la construction de l'anneau, l'attribution des clés aux nœuds, la découverte de la valeur et du nœud responsable, d'une clé donnée. Pour ce faire, les adresses (nœuds et ressources confondues) sont assignées dans l'espace [0 2^m-1] en utilisant un hachage cohérent (la probabilité d'une collision est très faible). L'algorithme SHA-1 est la fonction de hachage la plus souvent choisie.

Connaissance de la topologie par un nœud Chord
Un nœud *n* du réseau Chord détient une connaissance partielle de cette topologie à travers quatre éléments :

- son successeur, le nœud dont l'adresse suit immédiatement celle de *n* sur l'anneau ;
- son prédécesseur, le nœud dont l'adresse précède immédiatement celle de *n* sur l'anneau ;
- sa *Finger* table qui recense quels nœuds contacter pour atteindre m ressources de la topologie. Si on note Finger[k](*n*), il s'agit alors du nœud en charge de l'information $(n+2^{k-1})$ mod 2, avec $1 \leq k \leq m$, *n* l'identité de nœud, et *m* le nombre de bits pour créer l'espace d'identifiants. Plus la ressource d'une adresse est éloignée, moins le nœud *n* a d'information sur sa réelle localisation ;
- son index, l'ensemble des clefs dont le nœud est responsable.

Insertion d'un nouveau nœud

L'insertion d'un nouveau nœud, comme son départ, sont les deux événements majeurs de l'architecture. Pour que l'indexation et la recherche soient opérationnelles il faut s'assurer que tout nœud a la bonne connaissance de la topologie en fonction de son adresse. Au moment de son arrivée dans le réseau (*join*) un nœud doit trouver qui est son successeur. Pour cela, il lui suffit d'interroger le réseau pour trouver quel nœud est pour le moment responsable de son identifiant. A partir de là s'enchaîne la mise-à-jour de son prédécesseur, la mise-à-jour de l'index du nœud et de son successeur et la constitution de la *finger table*. Ces étapes sont expliquées plus en détail dans l'annexe *A.* De plus pour maintenir l'architecture, chaque nœud doit périodiquement vérifier son prédécesseur et son successeur (*stabilized*) et la validité de sa table de routage (*Fix Finger Table*).

Extention de Chord à la recherche par mot-clé

DHT P2P s'appuie sur le résultat de la fonction de hachage sur le nom précis de la ressource pour effectuer la recherche. Lorsque celle-ci est disponible, la recherche s'effectue rapidement et sans problème. Cependant, dans le cas où il n'existe qu'une information partielle sur la ressource, la phase de recherche échoue.

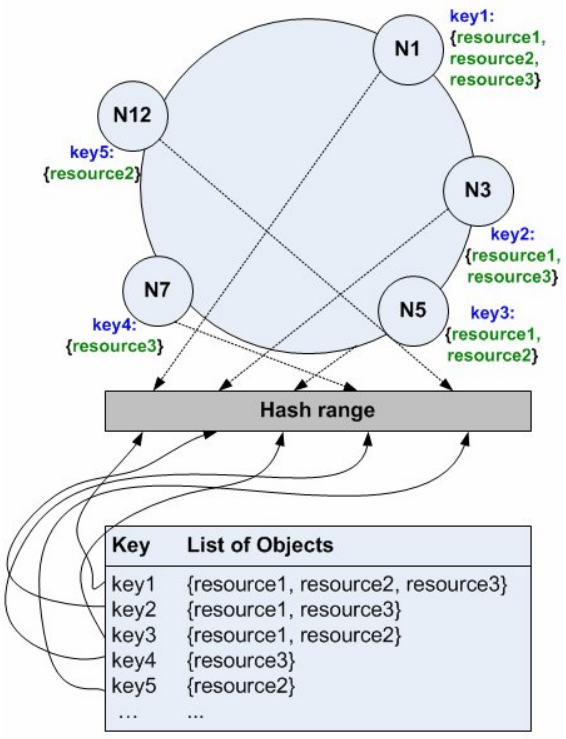

Figure 1.6.2-5: Distribution de l'indexation inversée aux nœuds Chord

Une solution courante pour résoudre ce problème consiste à mettre en place un *Index Inversé* [ReVa02][JoFY07]. Celui-ci est composé d'un ensemble de paires (*k*, *Res*), où *k* est un mot clé, et *Res* est un ensemble des ressources contenant ce mot-clé. L'index inversé d'une ressource est construit à partir de tous les mots-clés qui permettent de l'identifier. Le réseau P2P n'a pas à stocker l'index dans un seul nœud, l'index est

distribué comme pour le cas classique d'indexation dans une DHT (Figure 1.6.2-5), c'est juste qu'il faut stocker plusieurs ressources et faire plusieurs recherches pour chacun des mots clefs. Dans cet exemple, la recherche à partir des mots-clés *key1*, *key2* et *key3*, donne comme seul résultat la ressource 1.

Cependant, cette solution n'est pas toujours efficace, particulièrement en cas de recherche de mots clefs très communs, entraînant une utilisation abusive de la bande passante [SaGG02]. Pour résoudre ce problème, une première idée est de générer des clefs regroupant directement plusieurs mot-clefs : le Keyword-Set Searching System (KSS) [Gnaw02], mais cette solution est très coûteuse en stockage.

1.6.3. Intégration SIP et P2P

Nous avons vu d'une part les architectures centralisées et d'autre par les architectures distribuées. Si l'on cherche à situer, SIP nous pouvons indiquer qu'il se situe entre les deux modèles, puisqu'une fois la signalisation de session terminée, la communication a lieu directement entre les deux agents utilisateurs (P2P). Il parait alors naturel de tenter d'intégrer SIP au P2P, ces deux systèmes ayant pour objectif commun d'enregistrer et de localiser des données (si on considère que l'utilisateur SIP est une donnée).

Deux approches ont été proposées pour intégrer SIP et le pair à pair : SIP sur P2P et P2P sur SIP.

L'objectif de *SIP sur P2P* est d'améliorer le réseau SIP en éliminant (ou au moins réduire) le besoin de serveur centralisé. Un groupe de travail nommé P2PSIP a été crée à l'IETF [BMSW08]. À ce jour, le travail du groupe semble s'être arrêté et aucun RFC n'a été publié.

Quant au but du *P2P sur SIP*, il est de réutiliser SIP comme protocole de communication entre pairs. Par exemple, la téléphonie sur Internet peut être réalisée avec une architecture P2P où les participants forment un réseau auto-organisé qui offre un service de présence et de voix [SiSc04]. SIP peut aussi être utilisé comme protocole dans la réalisation d'une partie des fonctions d'une DHT.

Nous illustrons les deux approches d'intégration susnommées à la suite.

1.6.3.1. SIP sur P2P

Le service de localisation SIP est remplacé par un système P2P, SIP s'en servant pour la découverte de la localisation de son interlocuteur (Figure 1.6.3-1). Les informations indexées par le réseau P2P sont alors les URI courantes des agents utilisateurs indexées par leur URI publiques. La couche au dessus du réseau (notée overlay sur la figure) propose aux pairs de rejoindre et de quitter le réseau tout en pouvant en être que client.

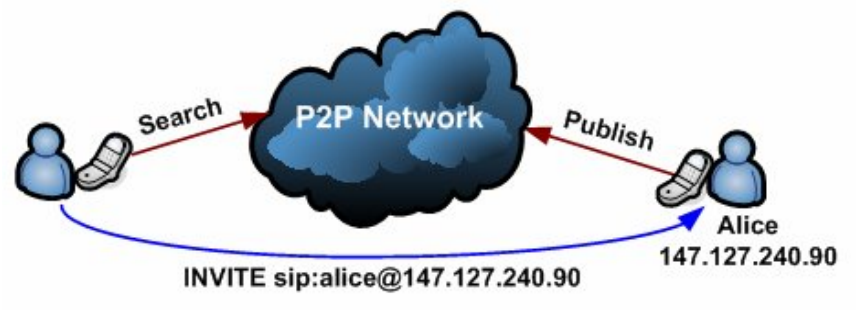

Figure 1.6.3-1: Principe de P2PSIP

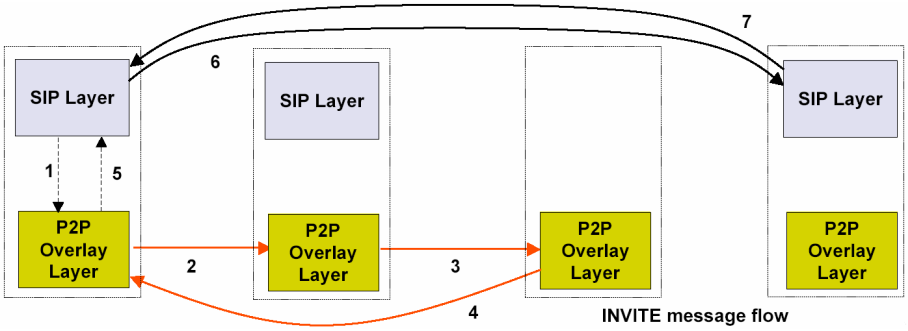

Figure 1.6.3-2: Le service SIP sur un réseau d'overlay [ShND06]

Un message SIP est transmis par une couche SIP à travers des overlay P2P pour rechercher l'URI courante de l'utilisateur SIP. Le pair P2PSIP indexant cette relation répond alors à la couche SIP (Figure 1.6.3-2).

1.6.3.2. P2P utilisant SIP comme protocole de communication

Dans le cadre d'un réseau P2P sur SIP, les communications de l'overlay sont mises en œuvre en utilisant le protocole SIP comme illustré dans Figure 1.6.3-3. Les messages SIP REGISTER permettent à un nœud de rejoindre le réseau, tandis que les messages INVITE permettent de mettre en œuvre le service offert par le réseau P2P, par exemple de la voix, du partage de fichier. La principale différence avec le cas précédent, c'est que SIP est ici le protocole de signalisation du réseau, si le service final est de la voix, c'est donc d'autant plus efficace.

Figure 1.6.3-3: Illustration d'un réseau P2P construit sur SIP

1.7. Conclusion

Nous avons présenté dans ce premier chapitre la notion de Home Service et de Home Network. L'accent a été porté sur la notion de communication entre domiciles au travers d'un état de l'art des architectures réseaux pour la gestion de service. Il apparaît que chacune de ces architectures offre de réelle capacité à déployer un service.

Dans la suite de ce document nous allons nous préocupper de préciser l'interconnexion des domiciles en prenant en compte des besoins de propriété des données et de sécurité des accès.

La propriété de l'information est une préoccupation de plus en plus prégnante. Dans Internet cette propriété est liée au stockage. Les serveurs et autres concepts semblables d'Internet (Cluster, Cloud, etc…) ont accès à l'information parfois en partie, mais souvent totalement. Ils la stockent et elle leur appartient. Mais pour le fournisseur de service, cela le rend responsable de ce qu'il stocke et pour l'utilisateur, il y a une perte de confidentialité et de ses droits. C'est peut-être l'une des raisons qui font prédire la chute des acteurs comme Google et Facebook à certains analystes [Guil12]. Pour aller plus loin sur ce point, l'indexation même de l'information par l'opérateur de services peut être considérée comme une perte de confidentialité. S'il est possible à l'opérateur d'offrir aux clients la gestion directe de leurs services et l'indexation de leurs ressources, cela pourrait ouvrir la voie à une nouvelle vison du web où les traitements ne seraient plus entrepris par le réseau mais considérés comme des applications (éventuellement mobiles) exécutées par l'utilisateur.

L'autre besoin dont nous nous préocuppons dans ce travail est essentiel, il s'agit de la sécurité. Ouvrir les HN à l'extérieur ne doit pas être une source d'intrusion supplémentaire. Les enjeux de la sécurité dans ce cadre sont alors de plusieurs natures. Tout d'abord il y a la préservation des données privées. Les données du client peuvent très bien être partagées avec sa famille, des amis mais il ne faut pas que celles-ci soient accessibles à n'importe qui. D'autre part, si le HN est le moyen de gérer un certain nombre d'équipements du domicile (climatisation, volets roulants, lumières, etc…) il n'est pas envisageable que le contrôle de ces équipements puisse tomber dans les mains de personnes autres que les propriétaires. Il faut donc s'assurer que les ressources du domicile et des services ne soient accessibles que par les personnes ad hoc.

En conséquence, nous avons besoin d'architecture réseau intégrant des mécanismes de sécurité. Dans la suite de cette thèse nous allons proposer et analyser des architectures à même de répondre à ces besoins. Pour ce faire nous allons nous appuyer sur un service de référence qui sera utilisé pour comparer différentes propositions d'architectures ainsi que pour valider par expérimentation nos propositions. Nous considérons également une architecture de référence, l'architecture IMS que nous décrivons brièvement pour terminer ce chapitre consacré aux éléments de base de la thèse.

1.7.1.1. Description du service de référence pour l'étude

Le scénario de service de référence que nous prenons est le service de partage de photos. Les photos sont des photos privées dont les droits d'accès sont ouverts à tout membre du réseau inter-domicile. Le service de partage de photos a été choisi en raison de sa simplicité, son coté pratique et le fait que son modèle de fonctionnement correspond à celui de services plus complexes

Détaillons ci-après les étapes de livraison du service et le diagramme d'activité de celui-ci.

Bob aimerait voir les photos de ses amis (membre de la communauté). Il démarre son dispositif client (par exemple, son téléphone mobile, son iPhone ou sa tablette tactile) pour contacter la passerelle de son domicile (HGw). Il écrit quelques mots clés pour rechercher des photos. Cette recherche peut être une requête à ses amis ou à des sites de partage de photos. Si la requête est fructueuse, il vérifie la disponibilité, les droits d'accès, et la compatibilité des équipements d'affichage. Après cela, une liste de photos est proposée et il peut choisir les photos dans cette liste. Il sélectionne alors son choix, télécharge et affiche certaines photos sur son périphérique client. Enfin, il peut éventuellement proposer celles-ci en partage.

Pour la suite de l'étude nous appellerons source, celui qui partage la photo, et consommateur, celui qui la télécharge.

1.7.1.2. Etapes de livraison de service

Le service de partage de photos est constitué de quatre étapes principales :
- l'enregistrement : les utilisateurs du service se connectent au gestionnaire du service et informe de leur statut ;
- la publication : la source indique au service qu'il partage une photo. La publication est donc l'étape propice à l'indexation de la ressource ;
- la recherche : le consommateur cherche les adresses des photos qui l'intéressent ;
- le rapatriement : cette phase consiste à récupérer la photo sélectionnée sur la source et à la rapatrier au consommateur.

1.7.1.3. Diagramme d'activité de service

Le diagramme d'activité de haut niveau pour le service de partage de photos est décrit dans la Figure 1.6.3-1. Le client souhaite obtenir une photo; le système recherche et affichage les résultats de recherche, puis, propose aux clients des liens pour les photos. Finalement, le client sélectionne un lien photo qu'il aime. Le système télécharge et affiche la photo.

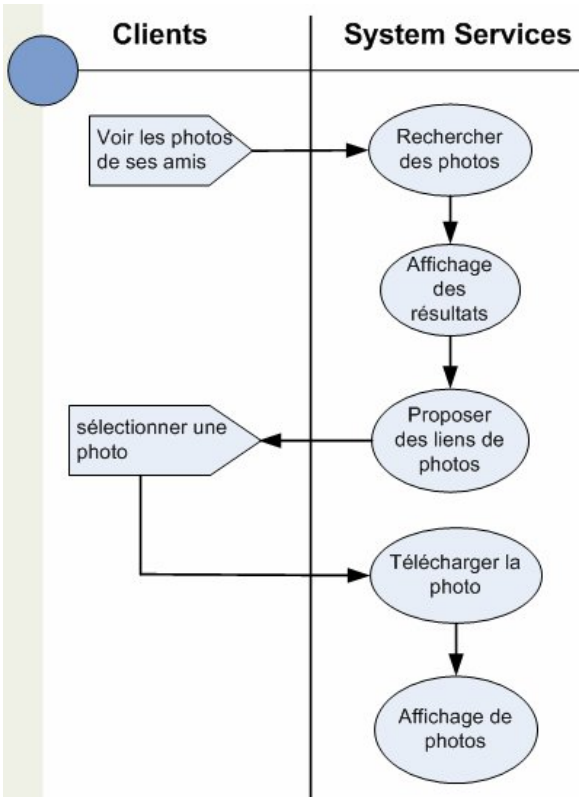

Figure 1.6.3-1: Diagramme d'activité du service de partage de photos

1.7.2. Architecture de référence pour l'étude

De même qu'il est utile de disposer d'un service de référence, une architecture est aussi nécessaire en vue de déterminer le domaine d'interêt d'une solution d'architecture devant une autre solution. C'est l'architecture de service opérateur qui nous semble toute désignée pour ce rôle : IP Multimedia Subsystem.

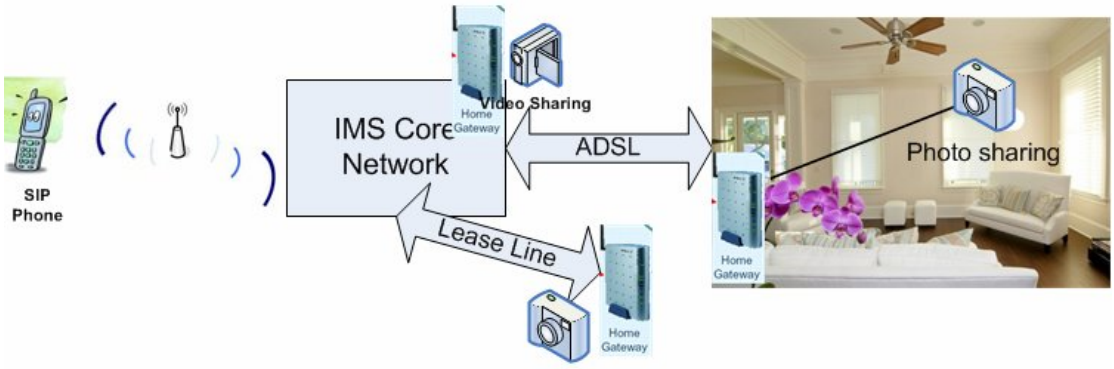

Figure 1.7.2-1: IMS comme support du service de partage de photos

IMS offre une architecture capable de fournir des services à domicile pour les clients de l'opérateur avec le fonctionnement suivant. La Figure 1.7.2-1 illustre la

connectivité offerte par IMS. L'équipement de l'utilisateur (par exemple, téléphone SIP) se connecte via le cœur de réseau IMS à son domicile pour utiliser le service de partage de photo de sa HGw. IMS prends en charge toute la gestion des identifications, autorisations, de connexion et de QoS. Toutefois il faut que ce service soit connu du système. La manière la plus simple est de créer un AS s'occupant de recenser les photos partagées pour le service de référence, un index central (Figure 1.7.2-2).

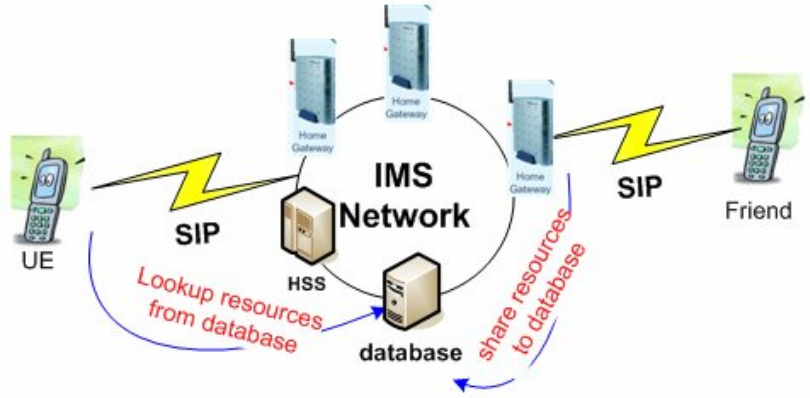

Figure 1.7.2-2: Utilisation d'un AS centralisé pour indexer les ressources du HS dans IMS

Chapitre 2

2. Mécanismes d'authentifications

Sommaire

2.1. Introduction

L'objectif de ce chapitre est de mettre en avant les mécanismes d'authentification applicables au contexte de l'interconnexion des réseaux à domiciles.

Notre objectif est d'utiliser ces mécanismes pour élaborer des architectures de délivrance de services authentifiées. L'intérêt d'une architecture devant une autre architectures est à considérer en regard de ses vulnérabilités de l'architecture réseau afin d'en caractériser les mécanismes les plus adéquats, mais également de son coût. En ce qui nous concerne, celui-ci est lié à la consommation de ressources réseau et dans ce travail nous n'expertisons pas les ressources de traitement. Nous nous attacherons à relever les échanges de signalisation de différents mécanismes et indiquerons simplement sans les détailler les algorithmes de cryptage qui sont exécutés sur les machines.

Nous commençons par présenter les concepts de base de la sécurité que nous utilisons dans cette étude. Ensuite, nous étudions les vulnérabilités de l'architecture de service en se concentrant sur le protocole SIP que nous avons choisi d'éudier en raison de sa forte popularité dans les services Internet, de son intégration dans les réseaux NGN avec IMS et des travaux en cours le concernant dans les réseaux pair à pair. Par la suite, nous présentons plus précisément 4 mécanismes d'authentification avant de nous intéresser à l'implantation de solution.

2.2. Notions d'authentification

Abordons en premier lieu l'abécédaire de l'authentification.

2.2.1. Cryptographie et clés

Afin d'effectuer une authentification, il faut s'appuyer sur des mécanismes de cryptographie. La cryptographie est l'outil de base de l'authentification. Elle est classiquement divisée en cryptographie à clés symétriques et clés asymétriques :

La cryptographie symétrique

Une clé symétrique est une information secrète. La cryptographie utilise souvent la cryptographie à clé identique pour à la fois le cryptage de texte en clair et le décryptage de texte chiffré. L'expéditeur et le destinataire doivent convenir d'une méthode pour crypter / décrypter un message. La clé, en pratique, représente le secret partagé entre deux ou plusieurs parties. Par exemple, un expéditeur effectue son codage par un décalage à droite d'une lettre de l'information qu'il veut transmettre (Hello → Ifmmp) et le récepteur effectue le décodage en décalant d'une lettre à gauche l'information reçue (Ifmmp → Hello) pour déchiffrer le message. Ce secret (décalage à gauche) est appelé clé symétrique. La clé symétrique de chiffrement est également dénommée clé de session, clé secrète ou encore clé partagée.

Des algorithmes pour le chiffrement utilisant des clés symétriques sont par exemple l'algorithme historique, Data Encryption Standard (DES), qui utilisait des clés de 56 bits, ou moins obsolète l'algorithme 3DES (avec 3 chiffrements et plusieurs clés) standardisé pour la sécurisation des transferts PPP (RFC 2420).

La cryptographie à clé symétrique avec une longueur de clé appropriée est considérée comme assez forte pour protéger le message. Elle est largement utilisée dans l'Internet. Cependant, le problème de cette solution est la gestion de la clé : comment le récepteur a-t-il connaissance de la clé? si celle-ci lui est transmise comment s'assurer de sa validité et de sa pérénité.

La cryptographie asymétrique

Elle utilise des clés différentes pour le chiffrement et le déchiffrement. Une de ces deux clés doit être maintenue privée, elle est appelée clé privée, alors que l'autre peut être rendue publique (sa transmission au récepteur n'a pas à être sécurisée), d'où son appellation de clé publique. C'est aussi pourquoi ce type de cryptographie est également nommé cryptographie à clé publique. Un exemple de clé publique est l'identité de l'utilisateur, on parle alors de mécanisme à base d'identité (Identity Based).

Un exemple d'algorithme très utilisé pour crypter/décrypter les messages est RSA (du nom de ses concepteurs : Ron Rivest, Adi Shamir et Leonard Adleman,) qui repose sur le choix aléatoire de nombres premiers par les clients pour générer leurs clés (sa robustesse est liée à la difficulté de la factorisation en nombres premiers). L'algorithme RSA s'appuie sur des longueurs de clés importantes (au moins 1024), ces tailles peuvent être réduites, au prix d'une complexité de calcul accrue, par l'utilisation de fonctions elliptiques (proposée pour le mécanisme IBC présenté par la suite).

Ainsi, contrairement à la cryptographie à clé symétrique, la cryptographie à clé asymétrique ne nécessite pas d'échange sécurisé initial d'une, ou plusieurs clés secrètes entre un émetteur et un récepteur. Les algorithmes de codage décodage fonctionnent de telle sorte que, tandis qu'il est facile pour le destinataire de générer les clés publiques et privées et de décrypter le message en utilisant la clé privée, ainsi que pour l'émetteur de chiffrer le message en utilisant la clé publique, il est extrêmement difficile pour quiconque de deviner la clé privée en fonction d'une connaissance de la clé publique. Les algorithmes s'appuient sur des relations mathématiques qui n'ont pas de solution efficace.

Cependant, le chiffrement à clé asymétrique est considéré comme plus lent que le chiffrement à clé symétrique ; c'est pourquoi il n'est généralement utilisé que pour les échanges à haute sécurité tels les échanges de clés session.

Certificat de clé publique

Afin de s'assurer que la clé publique de cryptographie à clé asymétrique, appartient à une personne réelle, il est nécessaire d'y associer un certificat de clé publique également connu sous le nom de certificat numérique ou certificat d'identité. C'est un document électronique qui utilise une signature numérique pour lier une clé publique avec une identité et des informations comme le nom d'une personne ou une organisation, leur adresse, etc. Le certificat peut être utilisé pour vérifier qu'une clé publique appartient bien à un individu.

Exemple d'utilsation conjointe de mécanismes d'authentification : IPsec
Les trois mécanismes élémentaires que nous venons de citer sont généralement associés. Ainsi, la cryptographie par clé asymétrique est utilisable pour négocier des clés secrètes partagées dans un schéma par clé symétrique après avoir vérifié la validité du certificat de clé publique.

IPsec est un exemple de solution standardisée par l'IETF dans Internet et utilisée pour IMS comme nous le verrons par la suite, qui met en œuvre plusieurs mécanismes. Elle permet de transporter des données sécurisées sur Internet en établissant une connexion sécurisée. Pour ce faire une phase d'authentification des extrémités de la connexion à lieu avant de pouvoir échanger des clés puis de transférer de façon protégée les données. Les mécanismes de sécurité : clés de session, certificats, secret partagé sont négociables par protocole.

2.2.2. Processus d'authentification

Nous allons décrire les mécanismes d'authentification élémentaires que sont l'authentification des messages et l'authentification du serveur ; ceux-ci seront ensuite intégrés dans des solutions d'authentification plus complexes en particulier pour les mécanismes relatifs aux solutions reposant sur le protocole SIP que nous présenterons dans les sections suivantes.

Facteurs d'authentification
Les mécanismes d'authentification sont utilisés pour s'assurer que les utilisateurs d'un service sont réellement ceux qui sont identifiés. La façon de faire l'authentification peut être divisée en trois catégories selon les facteurs employés (qui peuvent être combinés) :

- Les facteurs de propriété : quelque chose que l'utilisateur possède (par exemple, carte d'identité, jeton de sécurité, jeton de logiciel, téléphone ou téléphone cellulaire).
- Les facteurs de connaissance : que l'utilisateur connaît (par exemple, un mot de passe, une expression ou un numéro d'identification personnel (PIN), la réponse à un défi (l'utilisateur doit répondre à une question).
- Les facteurs intrinsèques: quelque chose lié intrinséquement à l'utilisateur de de manière unique (par exemple, des empreintes digitales, empreintes rétiniennes, la signature, le visage, la voix, ou tout autre identifiant biométrique).

En ce qui nous concerne nous considérons une authentification par facteurs de connaissance bien qu'il eut été envisageable de mettre en œuvre une passerelle réseau avec une authentification intrinsèque, mais il eut alors fallu considérer l'extension de cette authentification pour prendre en compte l'aspect ubiquitaire des services : l'utilisateur doit pouvoir accéder à distance à ses services au travers de divers dispositifs.

Authentification des messages
L'authentification des messages est un processus qui permet à un expéditeur et un destinataire de vérifier le message reçu. Ce processus peut garantir que le message ne

sera pas modifié (intégrité) pendant la transmission et confirmer qu'il est envoyé par l'expéditeur supposé (authentification).

La façon la plus simple consiste à chiffrer le message avec la clé secrète partagée. S'il y a un émetteur et le récepteur qui la connaisse, alors il n'y a qu'eux qui peuvent effectuer le décryptage permettant de lire le message. Ce qui assure une authentification du message. Par ailleurs, en ajoutant le numéro de séquence et un horodatage la sécurité peut être accrue. Il est alors possible de déjouer des attaques par rejeu (un échange capturé à un instant t et reinjecté à l'instant $t+x$ ne pourra pas être considéré comme bon).
Notons qu'une clé asymétrique peut également être utilisée pour faire l'authentification des messages.

Certains messages contiennent des informations qui peuvent être divulguées mais nécessitent d'être authentifiés. Une certification peut être nécessaire sans que le cryptage assurant la confidentialité des données ne soit requis. La clé est utilisée pour créer une empreinte de l'information à transmettre (message digest par exemple par fonction de hachage) que l'en émet en même temps que le message. Seuls les utilisateurs qui connaissent la clé secrète partagée peuvent digérer le message c.a.d. vérifier que le résumé du message reçu est correct (La clé peut être symétrique ou asymétrique)**.**

Authentification centralisée : L'objectif est de permettre aux utilisateurs de s'authentifier auprès d'un seul élément, le serveur. Tous les utilisateurs partagent leurs clés secrètes ou des certificats avec un seul serveur central. Trois méthodes sont envisageables dont les principes sont les suivants:

- Clé pré partagée avec cryptage: elle est utilisée pour émettre un message de demande d'authentification auprès du serveur. Ce message est crypté par la clé. Le serveur, s'il décrypte le message authentifie son émetteur. Il peut alors générer des clés de session qui seront transmises du serveur vers les utilisateurs en les cryptant par la clé pré-partagée.
- Clé pré partagée avec HTTP Digest: la clé pré-partagée est employée non pas pour crypter le message mais pour générer une empreinte du message (résumé) qui sera jointe au message. Le serveur, s'il décode le résumé, authentifie alors l'émetteur du message.
- Clé publique : l'utilisateur doit contacter une autorité de certification pour authentifier la clé publique de son correspondant.

Authentification Distribuée : dans cette option le recours à un serveur est un moyen complémentaire, il permet de stocker des informations ou alors d'amorcer le processus. L'évaluation de l'authentification est effectuée via un processus collaboratif entre amis (des utilisateurs authentifiés). La principale question à traiter est alors de savoir à quels nœuds du réseau l'on peut faire confiance. Selon les méthodes, les utilisateurs s'authentifient par une délégation de confiance auprès d'amis connus, des attributs de confiance

évalués par des amis, l'authentification de l'identité de l'utilisateur ou encore un processus d'accord Bysantin.

- Chemin de confiance ou Web of Trust: il s'agit pour un utilisateur d'authentifier la clé publique d'un autre utilisateur en la signant puis en la déposant sur un serveur, de telle sorte que la vérification de clé publique puisse s'effectuer par une recherche de *chemin de signature* : x a signé y qui a lui-même signé z, donc x peut authentifier z, la clé publique est bien la sienne (notons que différents niveaux de confiance peuvent être associés aux utilisateurs qui signent de telle sorte que des règles de confiance soient définies : si un utilisateur de niveau 1 signe alors c'est réussi, sinon il faut deux utilisateurs de niveau 2 …). Ce mécanisme a été initialement proposé dans la solution ouverte Pretty Good Privacy (PGP) [Zimm95].
- Score de confiance: l'authentification repose sur le calcul d'un score de confiance effectué par ses pairs en fonction du comportement des utilisateurs [WuHX08][ZhCY09].
- Cryptage de l'identité: comme son nom l'indique, le décryptage de l'identité de l'utilisateur permet de l'authentifier dans la mesure où la clé publique est l'identité de l'utilisateur. Avec cette méthode il n'y a pas de recours à un serveur. Notons cependant qu'il est nécessaire d'avoir des paramètres de sécurité complémentaires, générés à partir d'un serveur pour chiffrer et déchiffrer le message [Sham85].
- Authentification distribuée avec tolérance Bysantine : dans ce système il est possible que les amis qui interviennent ne soient pas de vrais amis (le système est Bysantin). L'authentification est réussie si une majorité d'amis honnêtes sont d'accord [PaIf06].

2.3. Examen de vulnérabilité

De façon à proposer une architecture d'authentification nous examinons dans cette section les problèmes qui doivent être résolus dans les deux modèles d'architecture de gestion de services : le modèle centralisé et le modèle distribué. Pour ce faire, nous analysons le protocole SIP [WABF09] car il est majoritairement utilisé comme protocole de signalisation dans l'établissement de session pour la délivrance de services (que ce soit dans les architectures IMS ou P2P SIP).

2.3.1. Les vulnérabilités de SIP

Globalement, nous classons les menaces de sécurité SIP en trois catégories. Tout d'abord, les attaques communes aux applications sur le réseau Internet telles que les attaques par rejeu, le déni de service, l'usurpation (spoofing), le reniflage (sniffing) et autres attaques par interception. Il y a ensuite des attaques spécifiques à la nature du protocole SIP (à la couche application). De par la structure de ses messages, SIP est vulnérable à certaines attaques telles que de fausses inscriptions ou terminaisons de sessions. Les vulnérabilités de SIP ont été relevées dès sa conception dans le RFC

3261 (celui-ci indique de façon générale les problèmes) mais sans qu'il y ait de solutions proposées. Finalement nous pouvons considérer, qu'en raison de la jeunesse et la complexité des applications reposant sur SIP (VoIP et le multimédia), les serveurs et autres produits SIP sont susceptibles de souffrir de vulnérabilités connues mais non forcément corrigées telles que l'injection SQL et le dépassement de tampon mémoire.

Des exemples d'attaques sont indiqués ci-après :

- *Attaque par rejeu*: *attaque contre la confidentialité*

C'est une attaque réseau dans laquelle une capture de données valides est malicieusement ou frauduleusement répétée (rejouée) en étant réinjectée dans le réseau par l'attaquant ou retardée. Cette attaque est réalisée soit par l'auteur ou par un adversaire qui intercepte les données et les retransmet (comme une attaque de chiffrement de flux en vue d'obtenir suffisamment de données pour pouvoir les décoder).

- *Phishing: attaque contre la confidentialité et l'intégrité*

Les attaquants peuvent usurper les identifiants utilisateur dans leurs connexions avec le serveur SIP pour voler des informations ou effectuer des actions malveillantes. Par exemple, l'attaquant peut enregistrer son proxy SIP - avec un nom semblable à un organisme de la victime - sur le site web proxy SIP de localisation. Lors du lancement d'une session SIP, la victime sera en réalité en contact avec l'attaquant du serveur SIP (au lieu de son serveur SIP légitime). L'attaquant sera alors en mesure d'obtenir des informations sur les victimes. Par ailleurs, l'attaquant peut usurper l'identité d'une entité SIP, par exemple l'identité de l'organisation de confiance et appeler la victime pour lui demander des informations afin de commettre acte un malveillant.

- *Voice trapping: attaque contre la confidentialité*

Comme le trafic de voix sur IP est envoyé sur le réseau IP sans aucun cryptage, l'espionnage des paquets est ainsi possible, en particulier dans les réseaux sans fil, où il est plus facile de piéger les données (par rapport au réseau téléphonique traditionnel qui est point à point, non diffusé).

- *Denial of Services (DoS): attaque contre la disponibilité*

SIP est vulnérable à plusieurs attaques DoS. En particulier, les attaquants peuvent forger de fausses adresses et envoyer plusieurs messages *INVITE SIP* générant ainsi de fausses ouvertures de sessions qui ne seront pas acquittées. Il est également possible de forger de fausses requêtes d'enregistrement, de modifier le champ « *Route Header* » etc.

- *Bogus terminate: attaque contre la disponibilité du service en cours*

Les attaquants peuvent émettre des messages BYE contrefaits ou effacer des messages au proxy SIP pour détruire des sessions en cours tel que illustré en Figure 2.3.1-1.

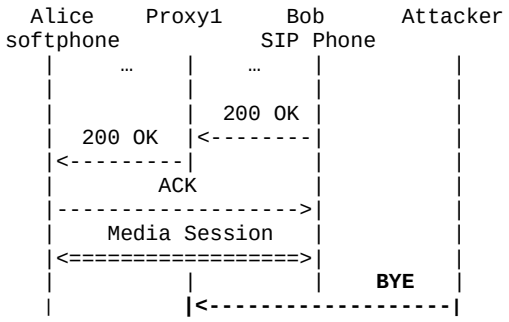

Figure 2.3.1-1: L'attaquant envoie un message BYE pour terminer la connexion.

- *Bogus register: attaque contre la disponibilité et la confidentialité*

Les attaquants peuvent émettre des messages *INVITE* malveillants vers le proxy SIP pour rediriger tous les messages de conversation destinés aux nœuds victime, et ainsi mener à bien une attaque de l'homme du milieu ou bien une attaque DoS.

- *Voice spam / Predicate call: attaque contre la disponibilité*

Les attaquants peuvent émettre des messages *INVITE* à plusieurs utilisateurs pour des sessions ennuyeuses c'est-à-dire avec un contenu propre à ennuyer l'utilisateur de la session SIP. Ce type d'attaque s'apparente à celles par pourriel (spam) tout en étant plus complexe à contrer. Il est possible de vérifier les emails qui ne sont pas en temps réel alors qu'il est plus complexe de vérifier en temps réel les signaux audio.

- *Message mal formé : attaque contre la disponibilité, la confidentialité et l'intégrité*

Pour exécuter des codes malveillants ou pour afficher des données confidentielles (par exemple, les utilisateurs de login / mot de passe), un attaquant peut provoquer un dépassement mémoire (Figure 2.3.1-2) ou une attaque par injection de code SQL (Figure 2.3.1-3).

```
INVITE sip:bob@enseeiht.fr SIP/2.0
Via: SIP/2.0/UDP 147.127.240.x:5060
From: %s%s%s%s%s%s%s%s%s%s%s%s%s%s <sip:alice@enseeiht.fr>; tag=765809
To: Bob <sip:bob@enseeiht.fr>
Call-ID: 8604553107@enseeiht.fr
CSeq: 1 INVITE
Contact: <sip:alice@147.127.240.x>
Content-Type: application/sdp
Content-Length: 138
:
```

Figure 2.3.1-2: Exemple d'un dépassement de tampon dans le cadre de la SIP.

```
Original register message:
REGISTER sip:sipdemo.enseeiht.fr SIP/2.0
From: Alice <sip:alice@sipdemo.enseeiht.fr>
:
SQL lookup: select * from users where username="alice";

SQL injection message:
REGISTER sip:sipdemo.enseeiht.fr SIP/2.0
From: Alice <sip:alice";drop table users;--@some.sip.domain>
:
SQL lookup: select * from users where username="alice"; drop table users;--;
```

Figure 2.3.1-3: Exemple d'une injection SQL dans le cadre de SIP.

Par ailleurs, l'interface web sur le serveur téléphonique central pour le téléphone SIP peut être vulnérable à une attaque par script croisé (cross-site scripting: XSS) qui résulte en une action ne correspondant pas à celle qui devrait être entreprise par la page web. Une attaque XSS permet à des attaquants distants d'injecter un script web malicieux, ou des codes HTML malicieux via le champ « *From Header* » dans un message SIP.

2.3.2. Les vulnérabilités de P2P-SIP

Après avoir illustré les attaques SIP étendons notre propos à l'architecture pair à pair. Dans une architecture P2P-SIP où le réseau pair à pair est un réseau superposé au réseau SIP utilisé pour l'enregistrement SIP et la gestion de l'emplacement de l'utilisateur, de nouvelles menaces de sécurité peuvent survenir. En fait, la collaboration de plusieurs nœuds non crédibles (c'est-à-dire qui ne sont pas authentifiés en tant que nœud crédible) ainsi que l'absence d'une autorité centrale introduisent plusieurs problèmes de sécurité, principalement liés à l'authentification et la confiance. Ainsi un attaquant peut émettre des contrefaçons de messages de routage dans un réseau afin que les données soient acheminées à des emplacements incorrects. Nous résumons ci-dessous les principales questions de sécurité en P2P-SIP.

- *Recherche et routage: attaque contre la disponibilité, l'intégrité et la confidentialité*

Au lieu de router un message selon le protocole d'overlay, tout nœud malicieux au sein de l'overlay peut déposer, modifier ou transférer un mauvais message [Seed06], par exemple avec une mise à jour incorrecte de table de routage Chord (*Finger Table*) [SMKK01]. Ce qui peut provoquer la divulgation d'informations, une altération malveillante, ou même une indisponibilité totale de la fonction de superposition.

- *Free Riding: attaque contre la disponibilité*

Dans tout système de pair à pair, on peut imaginer des situations où les nœuds utilisent les services, mais ne parviennent pas à fournir des services au réseau [AdHu00]. Dans un cadre P2P-SIP, on peut imaginer des situations où les nœuds utilisent le service d'enregistrement pour la superposition et l'emplacement, mais ne laissent pas passer d'autres messages. Une grande quantité de tels nœuds « *Free Ride* » finirait par aboutir à une dégradation ou une indisponibilité de la fonction de superposition.

- *Sybil attack : attaque contre la disponibilité*

Dans une attaque Sybil [Douc02], un attaquant subvertit le système de réputation d'un réseau pair à pair en créant un grand nombre d'entités sous un pseudonyme, puis les utilise pour acquérir une influence disproportionnée. La sensibilité d'un système de réputation à une attaque Sybil dépend du coût auquel les identités peuvent être générées, du degré auquel le système de réputation accepte les entrées provenant d'entités qui n'ont pas une chaîne de confiance les liant à une entité de confiance, et si le système de réputation traite toutes les entités à l'identique.

2.4. Etudes de solutions d'authentification

Les solutions d'authentifications que nous présentons s'appuient sur des technologies courantes conformément aux objectifs du projet Feel@home. Nous étudions quatre solutions d'authentification élémentaires 1) HTTP digest 2) Crythographie Basée sur l'Identité 3) Transport Layer Security (TLS)/Secure Socket Layer (SSL) et 4) l'authentification distribuée avec accord bysantin.

L'intérêt de la solution à base de HTTP Digest est sa simplicité, sa rapidité et le fait qu'elle est très largement utilisée actuellement. Quant à la cryptographie par identité, c'est une nouvelle génération de cryptographie asymétrique moins rapide que la solution précédente mais avec l'avantage de ne pas avoir recours de façon systématique à un serveur central. TLS/SSL est une autre solution largement répandue dans l'Internet qui est mentionnée comme exigence fondamentale de sécurité dans l'architecture P2P RELOAD que nous étudierons dans la suite de cette thèse. Finalement, la solution distribuée avec accord byzantin permet d'avoir une solution sans serveur.

Il est à noter que le premier mécanisme de sécurité a été défini pour HTTP et a été adaptée pour une utilisation avec le protocole SIP (RFC 3261). En raison de ses limitations relevées dans le RFC 2617 de nouveaux mécanismes tels TLS et secure MIME on été définis pour SIP (RFC 4346). Cependant HTTP digest est la solution obligatoire sur toutes les implantations et est très largement utilisée pour l'authentification des agents utilisateurs (SIP UA).

2.4.1. HTTP Digest

HTTP Digest a été spécifié par la RFC 2069 (une extension au protocole HTTP: Digest Access Authentification) et amélioré dans le RFC 2617.

Le RFC 2069 relève les types d'attaques susceptibles de perturber HTTP et spécifie un schéma d'authentification par mot de passe crypté traditionnellement maintenu par un serveur, générant une chaîne aléatoire (nonce) permettant de créer une empreinte. Le client génère une requête, le serveur renvoie un message avec un code erreur contenant un nonce. Le client crée alors une empreinte cryptée en utilisant des parties de la requête, du nonce reçu, et du secret partagé. Le client transmet la requête au serveur en y joignant cette fois l'empreinte. Le serveur effectue la même empreinte et authentifie positivement le client si le résultat calculé est similaire à celui reçu. (Ce mécanisme d'empreinte est également utilisable dans IPSec où les données sont transmises avec leur signature)

L'empreinte est issue d'une fonction de hachage (MD5 ou SHA1) qui a de forte chance de donner un résultat unique (2 messages différents ont de grandes chances d'avoir des empreintes différentes sinon il y a collision). La réponse d'authentification est constituée comme suit (où HA1, HA2, A1, A2 sont des noms de variables de chaîne):

$$HA1 = MD5(username:realm:password) \quad (2.4.1.1)$$

$$HA2 = MD5(method:digestURI) \quad (2.4.1.2)$$

$$response = MD5(HA1:nonce:HA2) \quad (2.4.1.3)$$

Par rapport à l'authentification de base de HTTP, le mot de passe n'est pas transmis en clair (dans ce cas il faut utiliser un transport crypté), il est inclus au travers d'une empreinte. Cela permet à certaines implémentations (par exemple, JBoss digestauth) de stocker l'empreinte HA1 plutôt que le mot de passe en clair. Le mécanisme HTTP digest peut être implanté via un serveur *RADIUS* pour générer des nonces ou pour le calcul d'empreinte [SSSW08].

Les calculs MD5 sont destinés à être « unique », ce qui signifie qu'il doit être difficile de déterminer l'apport original quand seule la sortie est connue. Si le mot lui-même est trop simple, cependant, il peut alors être possible de tester toutes les entrées possibles et de trouver une sortie correspondant (via une attaque par force brute, dictionnaire). En fait la fonction MD5 s'est avérée vulnérable et des méthodes pour résoudre la fonction de hachage sont indiquées dans [WaYu05]. Des améliorations sur la fonction de hachage permettant une meilleure protection ont été proposés tels le RFC 5843 : pour utiliser de nouvelles fonctions de hachage ou plus récent le RFC 6151 : pour améliorer MD5. Une autre amélioration de sécurité est proposée dans le RFC 2617 qui permet aux serveurs de mettre en œuvre des mécanismes de détection des collisions et des attaques par rejeu. Des attributs d'horodatage sont spécifiés en même temps que le nonce est généré. Les nonces récemment publiés sont également mémorisés pour éviter les attaques par réutilisation.

Le mécanisme HTTP Digest est destiné à remplacer l'authentification HTTP non cryptée qui est référencée comme étant l'accès de base dans le standard. Toutefois, il n'est pas destiné à remplacer les protocoles d'authentification forte, telle que l'authentification par clé publique. Bon nombre des options de sécurité du document RFC 2617 sont facultatives et si la qualité de la protection n'est pas spécifiée par le serveur, le client va fonctionner en sécurité réduite.

2.4.2. Cryptographie basée sur l'identité

La cryptographie symétrique utilise la même clé pour le chiffrement et le déchiffrement. C'est une technique simple d'utilisation et de réalisation, mais qui nécessite de régler des problèmes d'échange de clé. Tout d'abord, comment pouvons-nous échanger la clé secrète partagée en toute sécurité? Ensuite, pour communiquer avec n utilisateurs $O(n^2)$ clés sont nécessaires (une clé par couple de communication) ce qui augmente les possibilités d'interception, les délais de génération, les échanges…. Ces problèmes peuvent être éliminés en utilisant la cryptographie asymétrique qui ne nécessite pas d'échange de clé par session et pour laquelle seules 2 clés par utilisateurs sont nécessaires (O(n) clés). Cependant, subsiste toujours le problème de savoir comment nous pouvons nous assurer que la clé publique est

donnée à toute personne correctement. La solution proposée pour ce cas est d'utiliser des certifications émises par une autorité. La clé publique est alors vérifiée par un certificat. Cette technique est actuellement couramment utilisée mais elle nécessite de disposer de mécanismes pour obtenir et stocker les certificats.

En 1984, [Sham85] établit un schéma de chiffrement à clé publique dans lequel la clé publique peut être une chaîne arbitraire, comme l'adresse e-mail, l'adresse IP, l'adresse MAC etc. Ce schéma est repris dans [BoFr01] qui propose «Identity Based cryptography» (IBC). Il permet à n'importe quelle paire d'utilisateurs de communiquer en toute sécurité, pour crypter/décrypter, sans échange de clés publiques ou privées, sans mémoriser des répertoires principaux et sans utiliser les services d'une tierce partie. Un générateur de clé (PKG : Private Key Generator) publie une clé publique, la clé maître, qui permet en connaissant l'identité de l'émetteur de générer une clé publique. Pour obtenir la clé privée, la personne habilitée à utiliser une identité contacte le serveur PKG qui lui renvoie sa clé privée. Plusieurs schémas IBC sont présentés en [BoFr01], celui des auteurs repose sur un calcul par courbe elliptique pour générer les clés.

Un des avantages majeurs du système IBC est que les clés publiques sont dérivées à partir des identifiants, ce qui élimine le besoin d'une infrastructure de distribution de clés publiques. L'authenticité de la clé publique est implicitement garantie aussi longtemps que le transport des clés privées de l'utilisateur est maintenue sécurisé (authenticité, intégrité, confidentialité).
Toutefois, si un générateur de clé privée (PKG) est compromis, tous les messages protégés sur toute la durée de la paire de clés publique/privée utilisée par ce serveur sont également compromis. Afin de limiter l'exposition due à un serveur compromis, la paire de clés maître public/privé pourrait être mise à jour avec une nouvelle paire de clés indépendantes. Cependant, cela introduit un problème de gestion de clés où tous les utilisateurs doivent avoir la clé publique la plus récente pour le serveur. Un canal sécurisé entre un utilisateur et le Générateur de clé privée (PKG) est nécessaire. Une connexion sécurisée par TLS/SSL est alors une solution classique.

2.4.3. TLS/SSL

Le Transport Layer Security (TLS) qui est le successeur du protocole Secure Sockets Layer (SSL) fournit une protection pour la confidentialité, l'intégrité et la compression de données dans le cadre d'applications client/serveur. Il crypte les segments des connexions de réseau en utilisant 1) la cryptographie asymétrique pour l'échange de clés, 2) le chiffrement symétrique pour les messages et 3) des codes d'authentification de messages pour l'intégrité du message. Plusieurs versions sont largement utilisées dans des applications telles que la navigation Web, le courrier électronique, ou la voix sur IP (VoIP). Une utilisation importante de TLS est la sécurisation du trafic World Wide Web porté par HTTP pour le protocole HTTPS. Quand un serveur et un client communiquent, TLS assure qu'aucun tiers ne peut renifler les données et peut manipuler aucun message.

TLS [DiRe08] fonctionne dans la couche transport en encapsulant les protocoles d'application spécifiques, tels que HTTP, FTP SIP etc. Il est divisé en deux couches : TLS Record Protocol et TLS Handshake Protocol indiquées sur la Figure 2.4.3-1. TLS Record Protocol est responsable du cryptage des données à partir de données négociées par TLS Handshake Protocol pour chaque session.

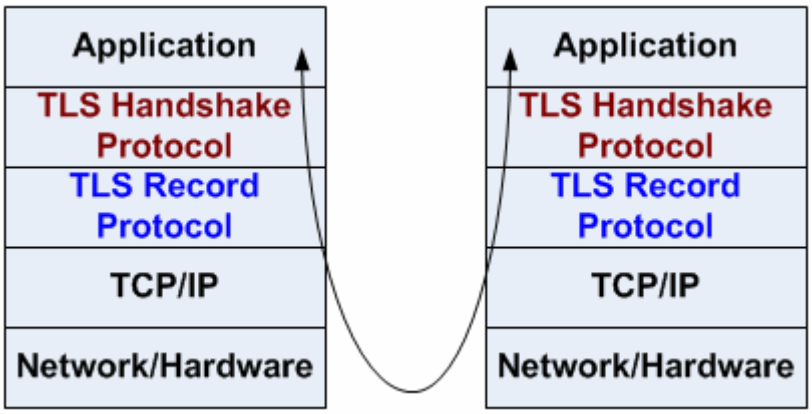

Figure 2.4.3-1: Les couches de TLS/SSL

Les principales caractéristiques du processus de sécurité sont :

Négociation du cryptage
Le client et le serveur prennent contact et choisissent la suite de chiffrement qui sera utilisée tout au long de leur échange de messages.

Authentification
En TLS, un serveur prouve son identité au client en utilisant le certificat. Le client peut également être amené à prouver son identité au serveur. Ceci est fait de base par un mécanisme de clé publique. La méthode exacte utilisée pour l'authentification est déterminée par la suite de chiffrement qui est négociée au cours de la période de Handshake.

Echange de clés
Le client et le serveur échangent des nombres aléatoires et un numéro spécial appelé le *Pre-Master* Secret. Ces chiffres sont combinés avec des données supplémentaires permettant au client et au serveur de créer leur secret partagé, appelé *Master secret.* Le master secret est utilisé par le client et le serveur pour générer la clé de session (*MAC secret*), utilisée pour le hachage et le chiffrement.

Établissement d'une session sécurisée en utilisant TLS
En nous référant au traçage des messages échangés sur une implantation réelle que nous présenterons par la suite, nous obtenons les échanges suivant :

Le protocole TLS Handshake comporte les étapes suivantes:

1. Le client envoie un message "Client Hello" au serveur, avec une valeur aléatoire et les suites de chiffrement prises en charge.

2. Le serveur répond en envoyant un "Server Hello", avec également une valeur aléatoire qu'il a choisie. Il y joint son certificat pour l'authentification et peut demander un certificat au client. Ce message se termine par un "Server hello done".
3. Si le serveur a demandé un certificat au client, le client l'envoie. Le client crée un échantillon aléatoire *Pre-Master secret* et le chiffre avec la clé publique du certificat du serveur, puis transmet vers le serveur. Le serveur reçoit le *Pre-Master secret*. (Le serveur et le client génèrent le *Master secret principal* et les clés de session en s'appuyant sur le *Pre-Master secret*.)
4. Le client transmet au serveur une notifiaction"*Change cipher spec*" pour indiquer qu'il va commencer à utiliser les nouvelles clés de session pour le hachage et cryptage des messages. Pour finir, le client indique dans le message de signalisation "*Client finished*".
5. Le serveur reçoit "*Change cipher spec*" et passe en chiffrement symétrique en utilisant les clés de session. Pour finir, le serveur envoie "*Server finished*".
6. Client et serveur peuvent désormais échanger des données d'application sur le canal sécurisé qu'ils ont établi. Tous les messages envoyés du client au serveur et du serveur au client sont cryptés en utilisant la clé de session.

2.4.4. Authentification distribuée avec accord bysantin

Le mécanisme par chemin de confiance de PGP peut être considéré comme un mécanisme distribué. Cependant son utilisation est davantage appropriée pour de petits réseaux que pour de larges structures. En effet se pose alors le problème de comportement malicieux de la part des tiers de confiance qui signent les clés. Ce problème est abordé dans la solution « bysantine ».

Tous les mécanismes d'authentifications décrits précédemment nécessitent un serveur centralisé alors que celui-ci [PaIf06] peut éliminer ce besoin. Il repose sur des nœuds de confiance qui doivent être majoritaires sur l'ensemble des nœuds du réseau.

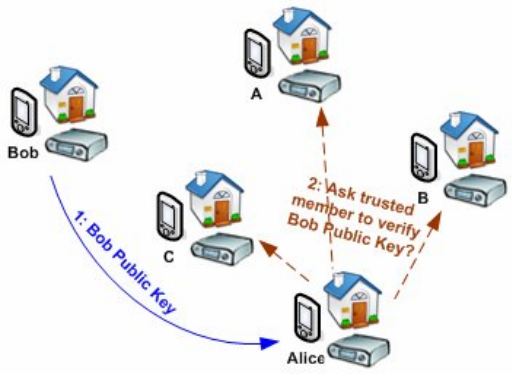

Figure 2.4.4-1: Vérification de la clé publique par membres de confiance-Etape1

Supposons que A, B et C (Figure 2.4.4-1) aient la confiance des nœuds dans le réseau, ils sont *nœuds de confiance* et fonctionnent comme des autorités de certification de façon décentralisée. Des nœuds de confiance peuvent être présélectionnés par un

administrateur pour amorcer le processus. Par la suite, les utilisateurs utilisent un mécanisme (accord byzantin) de détection des nœuds non de confiance. Les membres de confiance sont gérés de façon dynamique selon leur comportement.

Par exemple, Bob génère une paire de clés publique /privée. Quand il souhaite communiquer avec Alice, il envoie sa clé publique à Alice (Figure 2.4.4-1: flèche 1). Alice vérifie la clé publique de Bob en demandant l'aide de ses pairs de confiance (par exemple, A, B et C) (Figure 2.4.4-1: flèche 2). Les membres de confiance créent puis chiffrent un nonce avec la clé publique de Bob et signent le nonce avec leur clé privée (Figure 2.4.4-2: flèche 3). Ils l'envoient à Bob. Bob déchiffre le nonce pour chaque membre de confiance en utilisant sa clé privée et leur renvoie (Figure 2.4.4-2: flèche 3). Les pairs de confiance ayant décrypté le message contenant le nonce le transmettent à Alice (Figure 2.4.4-2: flèche 4). Alice peut valider la clé publique de Bob en comparant le nonce reçu des membres de confiance. Dans le cas d'une communication avec un intrus (par exemple un faux Bob), celui-ci ne peut pas déchiffrer le nonce s'il n'a pas la clé privée de Bob.

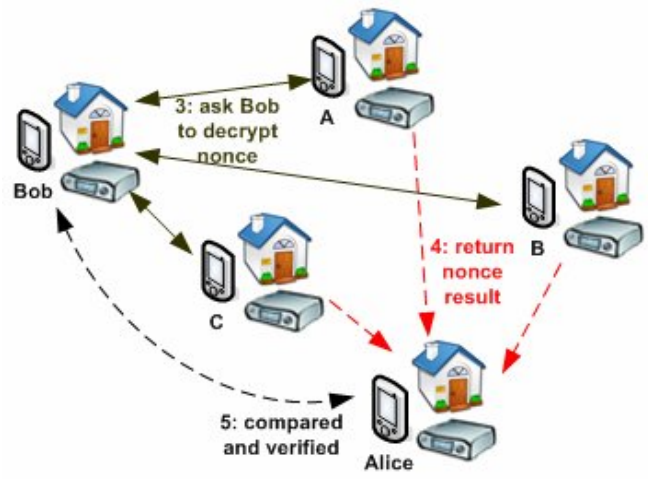

Figure 2.4.4-2: Vérification de la clé publique par membres de confiance-Etape2

Le résultat de l'authentification dépend de l'obtention d'un résultat consensuel des membres de confiance. Si le résultat n'est pas celui d'un consensus d'accord, Alice demande à Bob le résultat qu'il a préalablement émis aux nœuds de confiance, afin de détecter des pairs malveillants. Ces membres seront rattachés à un groupe « untrust ».

Une fois la clé publique de Bob authentifiée, Alice accepte sa demande et utilise la clé publique de Bob pour chiffrer et lui transmettre des requêtes. Bob peut également vérifier la clé d'Alice par le même processus.

Les principales fonctionnalités sont :

Pré requis
Un serveur de confiance conserve la liste des membres de confiance, ce serveur est de confiance pour tous les clients. Son adresse peut être pré configuré dans une

application cliente, avec une clé pré-partagée. Des membres de confiance peuvent être sélectionnés manuellement par l'administrateur réseau qui configure le serveur de confiance.

Amorçage

Quand Bob se connecte à un lien physique, il s'adresse à un serveur membre de confiance pour avoir les membres et les membres de confiance (flèche verte (1) en Figure 2.4.4-3). Le serveur lui renvoie une liste des membres de confiance (2). Cette communication est chiffrée en utilisant une clé pré-partagée avec Bob. Dans le réseau overlay P2P, chaque nœud a un lien direct pour se connecter au serveur de confiance. Bob diffuse sa demande pour trouver les nœuds les plus proches dans le réseau overlay P2P (3). Supposons que Alice est le nœud le plus proche de Bob, ils démarrent alors un processus d'authentification mutuel (4). Le processus d'authentification est exécuté avant qu'une connexion soit établie entre Bob et Alice. Bob a directement accès à tous les membres de confiance et à ses plus proches voisins. (Il est à noter que le choix des membres de confiance à contacter peut ne pas être les plus proches mais être choisis aléatoirement).

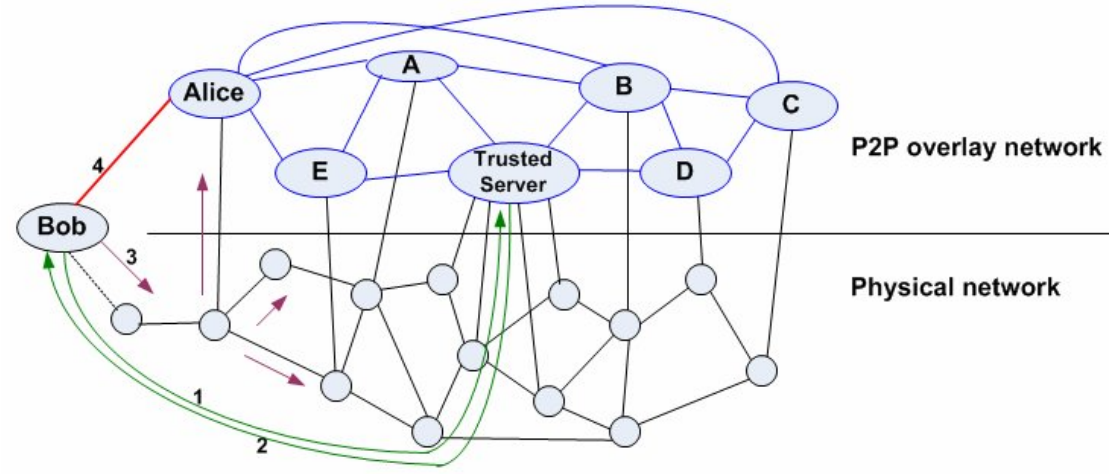

Figure 2.4.4-3: Processus d'amorçage

Processus d'authentification

Bob aimerait s'authentifier auprès d'Alice. Bob envoie sa clé publique et sa signature. Alice demande aux membres de confiance de vérifier la clé publique de Bob. Le(s) membre(s) de confiance (A sur la Figure 2.4.4-4) contacte Bob (ici par un mécanisme de question/réponse, challenge) et renvoie le résultat à Alice. Alice vérifie la clé publique de Bob en comparant le résultat qu'elle a obtenu à celui reçu de A. Si les résultats sont concordants, elle peut faire confiance à la clé publique de Bob. Par contre dans le cas contraire, elle doit s'adresser à Bob pour déterminer la présence d'éléments malicieux selon l'algorithme indiqué en Figure 2.4.4-5.

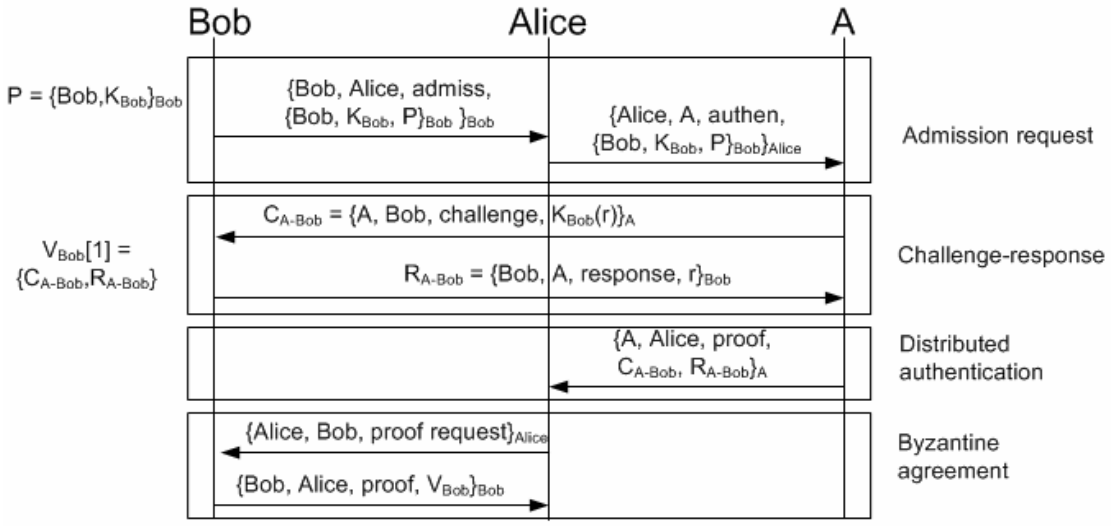

Figure 2.4.4-4: Echange d'authentification distribuée via un accord bysantin

Alice checks signature from $C_{A\text{-}Bob}$ and $R_{A\text{-}Bob}$
if ($C_{A\text{-}Bob}$ and $R_{A\text{-}Bob}$) from A are properly signed
  Alice checks "r" in $R_{A\text{-}Bob}$ by issuing $K_{Bob}(r)$,
    if "$K_{Bob}(r)$" is the same in $C_{A\text{-}Bob}$
      K_{Bob} is authenticated by A
    else
      goto **"Proof request"**
else
    There is a malicious peer or MITM at A (MITM= Man In The Middle)
Proof request:
      Alice sends request to Bob to get $V_{Bob}[1]$
        if ($C_{A\text{-}Bob}$ in $V_{Bob}[1]$ == $C_{A\text{-}Bob}$ in A) && ($R_{A\text{-}Bob}$ in $V_{Bob}[1]$ == $R_{A\text{-}Bob}$ in A)
          There is a malicious peer or MITM at A because r is not satisfied
        else if ($C_{A\text{-}Bob}$ in $V_{Bob}[1]$!= $C_{A\text{-}Bob}$ in A) && ($R_{A\text{-}Bob}$ in $V_{Bob}[1]$ == $R_{A\text{-}Bob}$ in A)
          There is a malicious peer or MITM at A because $C_{A\text{-}Bob}$ is signed by A
        else if ($C_{A\text{-}Bob}$ in $V_{Bob}[1]$ == $C_{A\text{-}Bob}$ in A) && ($R_{A\text{-}Bob}$ in $V_{Bob}[1]$!= $R_{A\text{-}Bob}$ in A)
          There is a malicious peer or MITM at Bob because $R_{A\text{-}Bob}$ is signed by Bob
        else if ($C_{A\text{-}Bob}$ in $V_{Bob}[1]$!= $C_{A\text{-}Bob}$ in A) && ($R_{A\text{-}Bob}$ in $V_{Bob}[1]$!= $R_{A\text{-}Bob}$ in A)
          Both of Bob and A are possible to be malicious peers or have some MITM

Figure 2.4.4-5: Algorithme de vérification de clef publique

Notation

K_{Bob}	Clé publique de Bob
r	Nonce aléatoire
$K_{Bob}(X)$	Une chaîne *X* cryptée par la clé publique de Bob
$\{X\}_{Bob}$	Un message *X* signé par la clé publique de Bob

Considération sur le processus d'authentification :
L'analyse présentée en [PaIf06] indique que le système proposé fonctionne si l'équation suivante est satisfaite :

$$t < \frac{1-6\theta}{3} n \qquad (2.4.4\text{-}1)$$

où t est le nombre d'intrus, n est le nombre de machines, θ est le nombre de chemins non fiables. Il faut que le nombre de message reçus justes soit plus important que le nombre de faux (n - 2t - 4θn > t + 2θn)

Illustration du processus d'authentification pour un service de partage de données

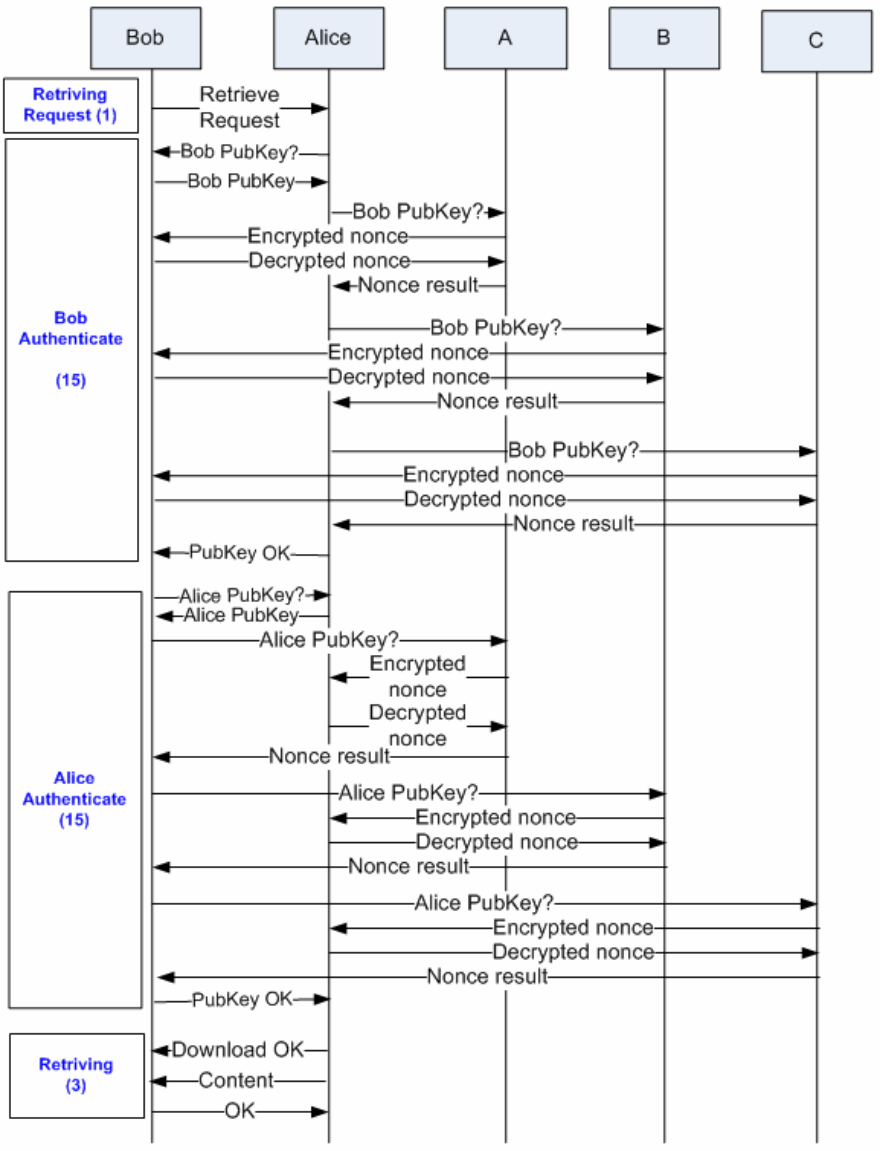

Figure 2.4.4-6: Processus d'authentification bysantin

Bob envoie une demande pour extraire des données stockées chez Alice. Alice demande la clé publique de Bob à Bob et aux membres de confiance. Ils transmettent une demande à Bob et retournent à Alice le résultat. Si les résultats mènent à un consensus sur la validité, la clé publique de Bob est vérifiée. Bob vérifie alors également la clé d'Alice. Bob et Alice peuvent communiquer en utilisant leurs clés publiques pour crypter / décrypter leurs messages ou bien ils peuvent les utiliser pour créer une autre clé secrète pour une meilleur sécurité. Cet échange est détaillé dans la Figure 2.4.4-6.

Inconvénients

- **Surcoût réseau:** la gestion des groupes de confiance et la gestion distribuée des clés génèrent un coût en termes de délai et de signalisation. Le réseau overlay auquel appartiennent les membres de confiance est un réseau maillé pour permettre de vérifier les clés publiques ; par ailleurs il est nécessaire de stocker sur les nœuds de confiance les clés publiques alors que dans un schéma classique seules les clés des utilisateurs avec lesquels un nœud souhaite communiquer sont stockées.
- **Surcoût de calcul:** le processus entraîne de nombreuses vérifications par session qui peuvent être inadaptées pour des dispositifs de faible capacité.
- **Complexité du contrôle des nœuds de confiance:** pour que le système fonctionne il est nécessaire d'avoir une majorité de nœuds de confiance.

En conséquence, il nous semble judicieux d'appliquer une authentification centralisée pour la prestation de services même si la gestion du service de partage est distribuée sur les utilisateurs pairs.

2.4.5. Considération pratique

Nous relevons que de nombreux services Internet sont utilisés actuellement avec une authentification centralisée complétée par des traitements distribués sur les utilisateurs :

- **Ebay:** c'est un site Web en ligne d'enchère et d'achats où les gens peuvent acheter et vendre une large variété de produits et de services. Les utilisateurs doivent s'authentifier avec un serveur centralisé par HTTPS. Chaque utilisateur peut évaluer le degré de confiance de l'utilisateur avec lequel il veut communiquer en fonction de son comportement passé : chaque vendeur est crédité d'un score de confiance par les acheteurs auxquels il a vendu quelque chose, de même chaque acheteur se voit attribuer un score de confiance.
- **BitTorrent:** c'est un protocole P2P de partage de fichiers utilisé pour distribuer de grandes quantités de données sur Internet. BitTorrent est l'un des protocoles les plus courants pour transférer des fichiers volumineux. Tous les utilisateurs doivent s'authentifier avec un serveur centralisé via HTTPS. Ils donnent leurs fichiers en affichant les adresses de partage des ressources (appelé fichier torrent) sur un site de traqueur web (web tracker). Le téléchargement s'effectue en mode P2P avec l'information à l'intérieur d'un fichier torrent.

- **Skype**: Skype est un service de transmission de voix sur Internet où tous les utilisateurs doivent s'authentifier avec un serveur centralisé en employant un protocole propriétaire dont les caractéristiques ne sont pas publiques.
- **Couchsurfing.org**: c'est un réseau pour mettre en contact des voyageurs avec des membres des communautés locales, qui offrent un hébergement gratuit et / ou des conseils. Tous les utilisateurs doivent s'authentifier avec un serveur centralisé via HTTPS. Les commentaires laissés par les utilisateurs servent également de score de confiance.

2.5. Implémentation de l'authentification

Après avoir présenté les mécanismes d'authentification qui peuvent être utilisés pour sécuriser l'architecture de services, nous nous intéressons plus précisément à leur mise en œuvre. Tout d'abord, nous indiquons une classification des mécanismes en fonction de leurs niveaux (par rapport à l'architecture de communication). En second lieu nous fournissons des précisions sur la mise en œuvre de la sécurité à partir de logiciels disponibles sur Internet.

2.5.1. Synthèse des mécanismes

Nous divisons les mécanismes de sécurité présentés précédemment, selon leur couche d'appartenance: la couche application, la couche transport, et la couche réseau comme le montre la Figure 2.5.1-1. HTTP Digest et IBC se situent dans la couche application. TLS gère l'authentification et le cryptage dans la couche de transport et IPSec gère la communication sécurisée dans la couche réseau.

HTTP Digest utilise généralement l'algorithme de hachage MD5, TLS et IPSec disposent d'options. RC4/3DES sont des algorithmes de clé de session alors que les algorithmes de IBC ou l'algorithme RSA relèvent de la classe clé publique/privée.

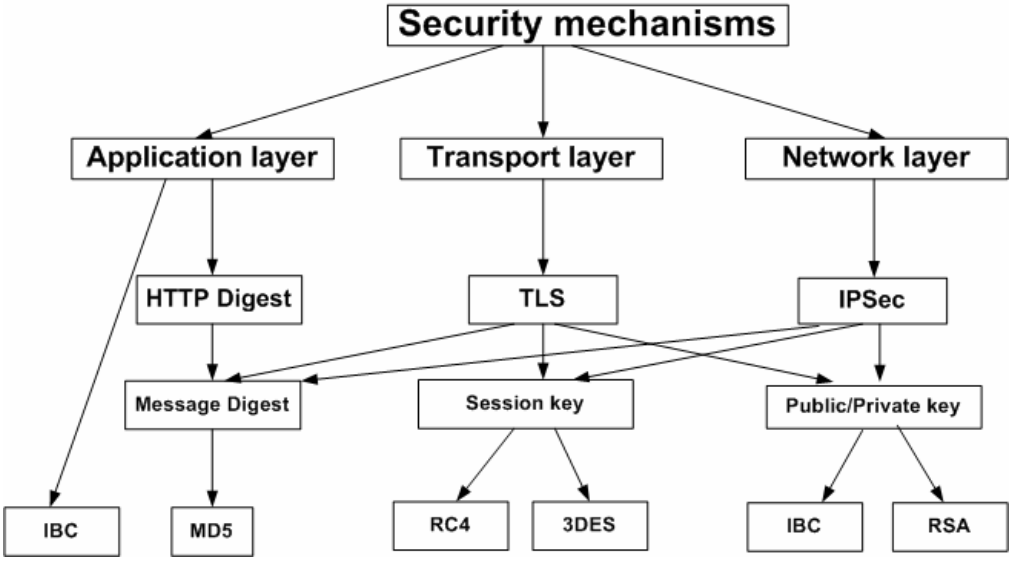

Figure 2.5.1-1: Mécanismes de sécurité par niveau

2.5.2. HTTP Digest

La norme SIP intègre le mécanisme HTTP Digest. Nous définissons pour sa mise en œuvre 2 interfaces SIP appelées *AuthenticationMethod* et *ClientAuthenticationMethod* comme le montre la Figure 2.5.2-1. Nous spécifions également les classes *DigestServer-AuthenticationMethod* et *DigestClientAuthenticationMethod* qui héritent des interfaces. Ces classes gèrent l'algorithme MD5 (par exemple, initialisation aléatoires, manipulation des caractères et des liens à digérer, processus de bas niveau dans java.security etc). Les noms d'utilisateurs et mots de passe sont fixés par *USER_AUTH* (username) et *PASS_AUTH* (password) variables. Realm (domaine) et nombre aléatoire (nonce / cnonce) sont également utilisés comme paramètres pour effectuer l'empreinte du message.

Nos objets HTTP Digest sont généralement utilisés avec une clé pré-partagée (en tant que paramètre de *PASS_AUTH*). Nous utilisons les méthodes *GenerateSignatur*e pour créer la signature du message et *VerifySignature* pour vérifier l'utilisateur.

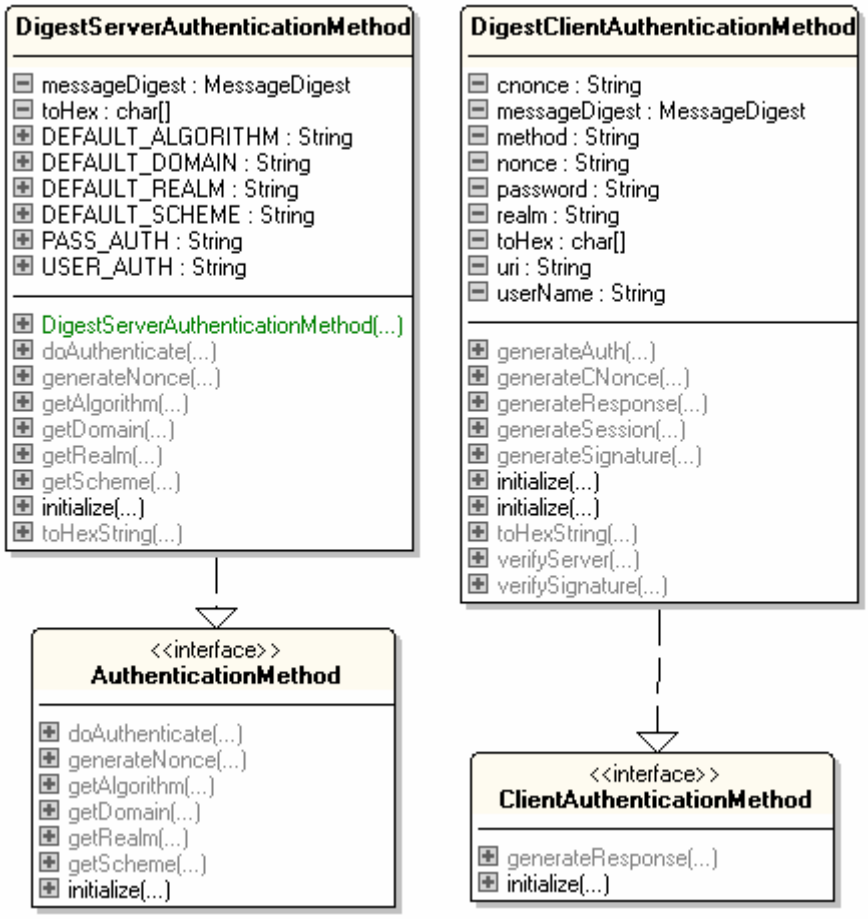

Figure 2.5.2-1: Diagramme des classes HTTP Digest

2.5.3. Identité Based Cryptography

Nous prenons la bibliothèque d'IBC du projet de jPair [Dong10]. Elle propose une application pure Java pour effectuer la forme bilinéaire d'appariement elliptique. Il n'y a pas de dépendances sur des bibliothèques externes. Il y a 3 interfaces principales : *Field*, *FieldElement* et *Pairing* comme le montre la Figure 2.5.3-1. Les 2 classes principales : BFCtext et BFCipher qui utilisent l'identité pour crypter et décrypter les messages sont illustrées en Figure 2.5.3-2. Nous avons testé le fonctionnement à l'aide d'une identité (par exemple, bob@enseeiht.fr) comme une clé publique pour crypter / décrypter et constater son bon fonctionnement.

Plus de détails sur la mise en œuvre à courbe elliptique sont disponibles dans le projet jPair [Dong10].

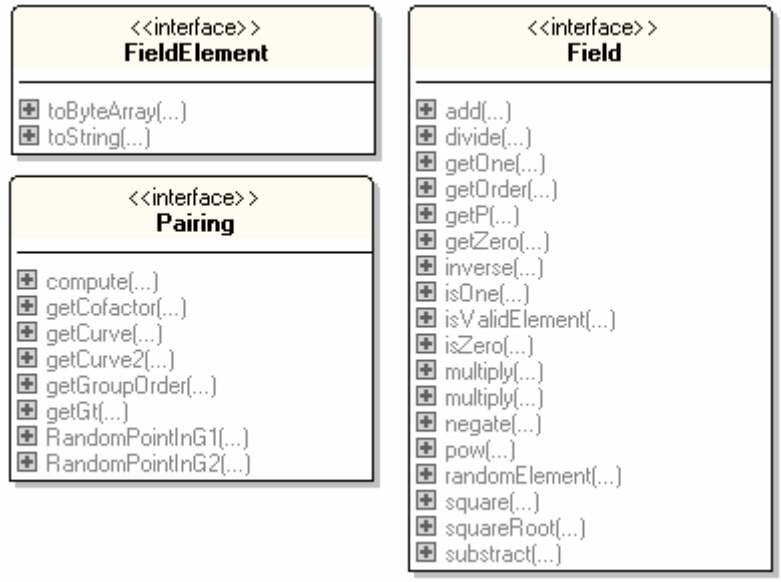

Figure 2.5.3-1: Diagramme des classes IBC

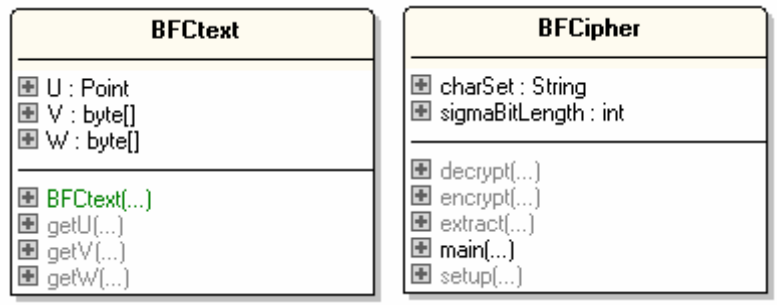

Figure 2.5.3-2: IBC class diagrammes

2.5.4. TLS/SSL

La cryptographie à clé asymétrique (clé publique / privée) est utilisée dans notre mise en œuvre de TLS avec des socket transport sécurisés *SSLsocket*. Les clés publiques et

privées sont respectivement appelées en Java *truststore* et *keystore*. La fragmentation et le réassemblage de message est fait en utilisant SSLSocket de la bibliothèque javax.net.ssl. *SSLContext* gère l'initialisation des objets *KeyManagerFactory* et *TrustManagerFactory*, le chargement dynamique *KeyStore* et *TrustStore* qui sont nécessaires dans SSLSocket.

Une fois que le *SSLSocket* est créé, nous avons besoin d'*inputStream* et d'*outputStream* pour l'envoi et la réception de messages aux extrémités de la pipe, *BufferedReader* est utilisée pour lire les messages du *SSLSocket*.

Nous définissons l'objet *NetworkManager* pour gérer les fonctions TLS/SSL (par exemple, les méthodes *acceptIncoming* et *sendMessage*) et créons la classe *ServerListener* pour l'écoute avec le SSLSocket (indiquée la Figure 2.5.4-1).

Dans le concept P2P, tous les nœuds sont serveur et client en même temps. Pour cette raison, ces objets sont exécutés en tant que « *Threads* » qui peuvent être tout à la fois en l'écoute et en exécution d'autres méthodes. Ainsi, dans l'étape d'initialisation, tous les nœuds doivent exécuter la méthode *acceptIncoming* pour écouter d'autres nœuds. Un nœud peut envoyer une demande à d'autres nœuds en utilisant TLS sur un autre port.

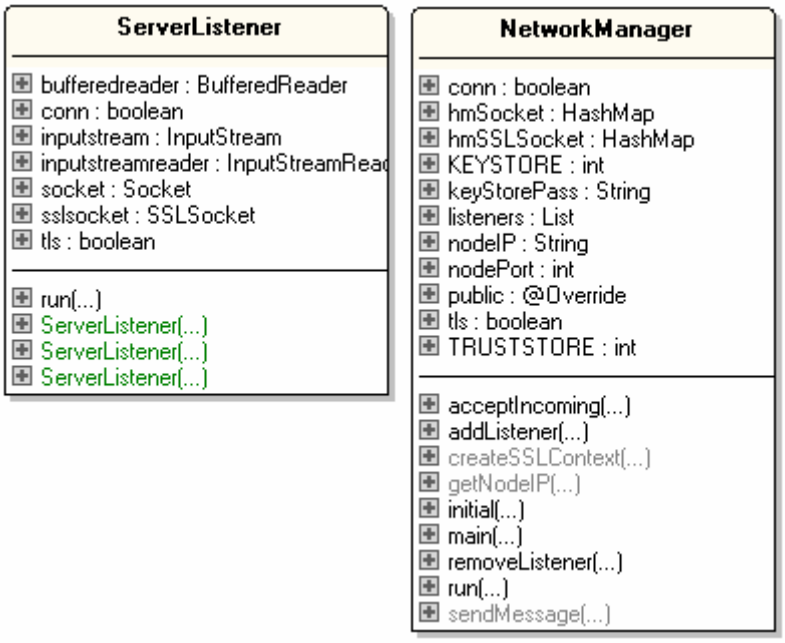

Figure 2.5.4-1: Diagramme de classe TLS/SSL

2.6. Conclusion

La sécurité est un point important qui doit être considéré lors du choix d'architecture pour la conception d'un système de délivrance de services au domicile. Elle peut être conçue de façon centralisée ou de façon distribuée. Les méthodes distribuées offrent éventuellement une diminution du délai pour les utilisateurs en cas de forte charge susceptible de pénaliser le processus d'authentification centralisé, ou encore une augmentation de la sécurité induite par la diversité des sources. En revanche, dans le cadre distribué se pose le problème de la détermination des nœuds de confiance. Il peut par exemple être supposé que tous les nœuds qui sont sous le contrôle de l'opérateur du système n'agiront jamais d'une façon malicieuse ou inexacte, dans ce cas tous les nœuds sont de confiance et l'architecture P2P peut atteindre un niveau de sécurité élevé, dans le cas contraire l'authentification distribuée est très complexe.

Dans la mesure où l'hypothèse d'un contrôle sûr de tous les nœuds par l'opérateur ne nous semble pas réaliste, nous optons dans la suite de ce travail pour une authentification centralisée, cette solution moins complexe que la solution distribuée semble également plus mature sur Internet.

Pour développer l'architecture de sécurité, les mécanismes de sécurité que nous avons identifiés dans ce chapitre sont IBC, HTTP digest, TLS et IPsec qui font intervenir des algorithmes de cryptage variés. Avec le point de vue adopté, l'algorithme de cryptage a moins d'impact sur le coût de l'architecture que les échanges générés. C'est pourquoi nous nous sommes attachés à décrire la signalisation induite par ces mécanismes. Au-delà des documents de référence de l'Internet, nous nous sommes appuyés pour ce faire sur des expérimentations.

Dans les chapitres suivants nous proposons d'utiliser ces mécanismes pour sécuriser une architecture de délivrance de services en pair à pair avec une gestion du service centralisée puis distribuée.

Chapitre 3

3. Architectures centralisées pour la délivrance de services au domicile

Sommaire

3.1. Introduction

Ce chapitre traite de l'authentification dans les architectures de service centralisées. Nous étudions notamment les architectures IMS (IP Multimedia Subsystem) et P2P centralisée pour des services de partage de contenu. Le concept de base d'une architecture centralisée est l'enregistrement de l'identifiant des machines possédant des ressource à partager en un seul endroit (central). La récupération des ressources partagées s'effectue par raccordement direct entre les nœuds, de domicile à domicile. Cette architecture offre de nombreux avantages en termes de rapidité de recherche et de mise en œuvre avec une gestion simplifiée par rapport à celle d'un système de gestion d'index distribué. Notons que l'inconvénient du passage à l'échelle inhérent à la centralisation peut être levé par une hypothèse de puissance suffisante.

Dans la première section, nous expliquons en détail les fonctions des architectures centralisées (IMS et P2P) en se référant au scénario de service de partage de photos. Les deux architectures reposent sur une signalisation par le protocole SIP. Dans la deuxième section, nous nous intéressons aux mécanismes d'authentification dans ces architectures. Dans la troisième section, nous effectuons des expérimentations avec le logiciel Open IMS et implémentons une architecture P2P utilisant nos propositions de mécanismes d'authentification. La quatrième section analyse et compare les mécanismes en terme de signalisation. Nous nous appuyons sur cette analyse pour proposer une architecture alternative à IMS, l'architecture SIP proxy.

3.2. Propositions d'architectures

3.2.1. IMS

Le fonctionnement de l'architecture IMS est illustré sur le service de partage de photos avec un serveur de catalogue central qui a été présenté dans le chapitre 1. Le serveur de catalogue est utilisé pour gérer les adresses des ressources qui sont publiées par les clients.

Dans ce scénario, nous utilisons deux domaines (ims.irit.fr et ims.enseeiht.fr) et trois utilisateurs (Bob, Alice et Eve). Sur la Figure 3.2.1-1, le réseau d'origine de Bob est ims.irit.fr alors que Alice et Eve sont sur le réseau ims.enseeiht.fr. Ils peuvent télécharger leurs listes de photos sur le serveur de catalogue. Le serveur de catalogue est directement relié au serveur d'application (AS) qui est aussi en communication directe avec les CSCFs. L'élément AS gère le service de partage de photos en utilisant comme base de données le serveur de catalogue. Un des rôles d'un CSCF est de traiter et de déterminer le prochain composant qui doit recevoir le message SIP. Lorsque le message provient d'un utilisateur, le choix du composant vers lequel orienter le message dépend du type de l'application: AS pour la recherche ou bien CSCFs du site pair pour le téléchargement.

Lorsque Bob souhaite accéder à ses photos depuis l'extérieur (dans notre exemple ims.irit.fr), il passe par le serveur de catalogue (après les étapes d'authentification et d'autorisation) pour obtenir des adresses privilégiées des photos sur CSCFs. Ensuite, il peut télécharger directement chez lui en utilisant les adresses récupérées. Les CSCFs

seront responsable de l'établissement de la session qui comprend l'authentification et l'autorisation au serveur de catalogue.

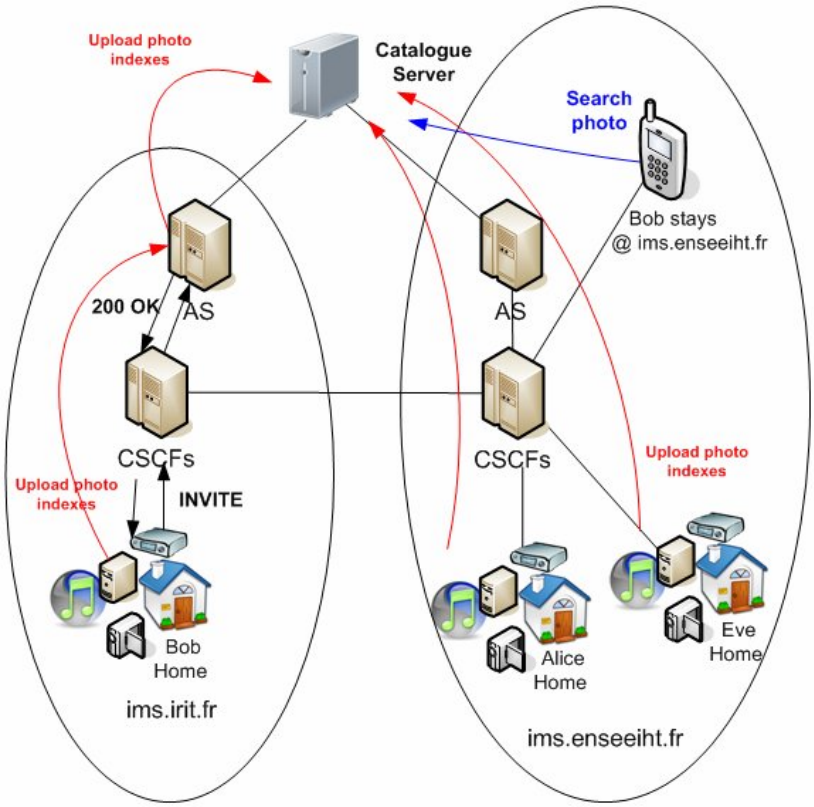

Figure 3.2.1-1: Illustration de l'architecture IMS pour le service de référence

3.2.2. P2P SIP centralisé

La seconde architecture proposée pour le partage de données est : P2P centralisée. Celle-ci présente l'avantage de reposer sur un produit ouvert, facilement gérable et compréhensible. Le protocole de signalisation est, tout comme dans l'architecture IMS, le protocole SIP [RSCJ02].

Scénario P2P: Le fonctionnement du scénario de service de partage de photos est similaire à celui présenté sur l'architecture IMS. Il y a publication, recherche et récupération. Les index sont stockés dans le serveur de catalogue central tel que décrit dans la Figure 3.2.2-1. Les utilisateurs connectés sur n'importe quelle entité dans le réseau peuvent partager en téléchargeant les indices photo sur le serveur de catalogue. Puis, un autre utilisateur (par exemple, Bob) peut effectuer des recherches et télécharger directement la photo souhaitée en se raccordant au domicile du propriétaire de la photo.

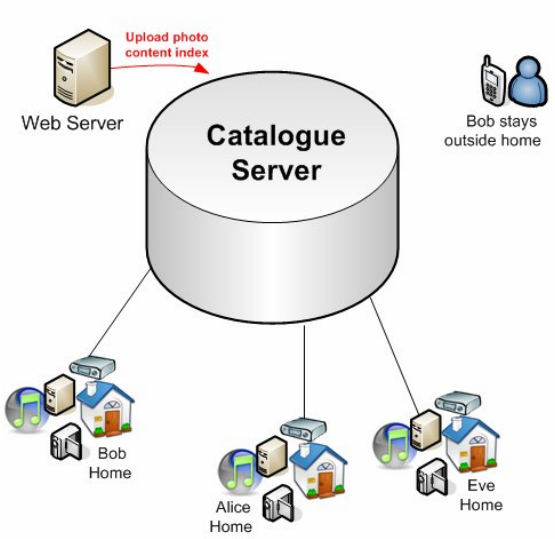

Figure 3.2.2-1: Illustration d'une architecture P2P centralisée pour le service de référence

3.3. Proposition d'authentification

Dans cette section, nous allons décrire les mécanismes d'authentification qui sont utilisés dans les architectures centralisées: IMS, puis nous proposerons des mécanismes pour le pair à pair.

3.3.1. Aspect sécurité IMS

La norme IMS propose une architecture de sécurité qui utilise trois protocoles de sécurité en fonction de la connexion à sécuriser (Figure 3.3.1-1):

- IMS Authentication Key Agreement (AKA) [NiAT02] entre l'équipement utilisateur (UE) et IMS réseau via le P-CSCF
- IP Sécurité (IPSec) [KeAt98] entre UE et P-CSCF
- Diameter [CLZA03] entre HSS et I/S-CSCF

L'architecture IMS de sécurité est complexe car elle nécessite les trois protocoles. C'est pourquoi elle est généralement simplifiée dans les environnements de tests, ce qui conduit à des vulnérabilités que nous présentons par la suite. L'analyse de ces vulnérabilités permet d'illustrer notre propos sur la sécurité mais dans la mesure où la sécurité implantée ne reflète par un réel système IMS, il est difficile d'utiliser l'expérimentation pour faire de l'analyse. Nous avons alors procédé par formulation (en comparer ses signalsations) comme nous le verrons en fin de chapitre.

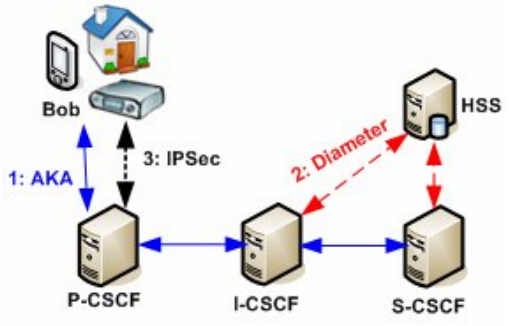

Figure 3.3.1-1: Vue globale de la sécurité dans d'IMS

Dans le plan de sécurité de l'IMS, il y a P/I/ S-CSCF et UPSF (User Profile Server Function, ou appelé HSS). P/I/ S-CSCF sont utilisés dans les procédures de sécurité IMS pour l'authentification entre l'équipement utilisateur et le cœur de réseau, pour accorder ou non l'autorisation de générer une nouvelle clé.

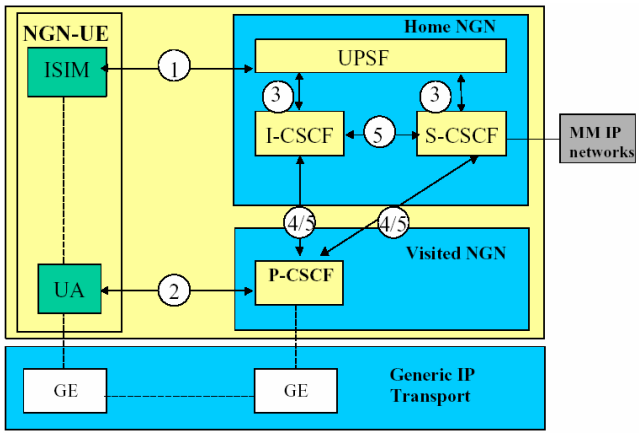

Figure 3.3.1-2: Architecture de sécurité d'IMS

La Figure 3.3.1-2 représente l'architecture de sécurité IMS avec un accès de transport IP pour se connecter à un réseau Multimedia IP (MM IP). L'IMS est indépendant du réseau de transport. L'élément noté GE (Generic Entities) dans le module Transport IP générique, est équivalent à des entités de transport 3GPP. L'équipement utilisateur (User Equipment) peut être soit un ISIM (IMS Subscriber Identity Module) ou un UA (User Agent) (client ayant une adresse IP). L'utilisateur ISIM-UE peut être connecté directement à Home NGN (numéro 1). Quant à l'utilisateur de type UA, il se connecte directement à un équipement P-CSCF (numéro 2) dans les réseaux NGN visités ou via le domaine du transport IP générique. Le module IMS Subscriber Identity (ISIM) qui est présent sur les téléphones portables utilise une clé pré-partagée. Lorsque ce module n'existe pas (cas des PC) il est possible d'utiliser un autre mécanisme tel que MD5.

Comme décrit précédemment à propos de l'architecture de sécurité IMS, quand l'équipement utilisateur souhaite accéder à un réseau géré par IMS, il doit effectuer

une authentification mutuelle qui est appelé AKA (Authentication and Key Agreement). IMS fournit des services d'authentification, la confidentialité et l'intégrité pour le chemin de signalisation SIP entre l'utilisateur et le réseau IMS (Figure 3.3.1-3). SIP utilise le mécanisme HTTP Digest pour l'authentification. Les paramètres AKA qui sont utilisés sont décrits dans la RFC 3310 [NiAT02].

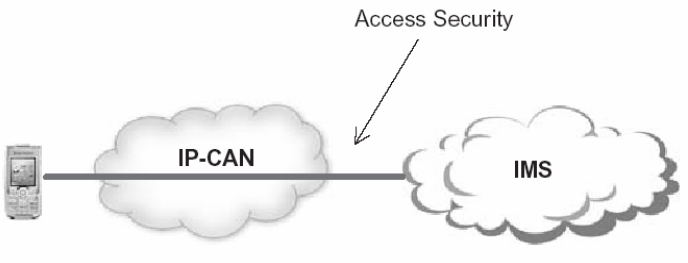

Figure 3.3.1-3**:** Accès sécurisé d'un utilisateur terminal à IMS

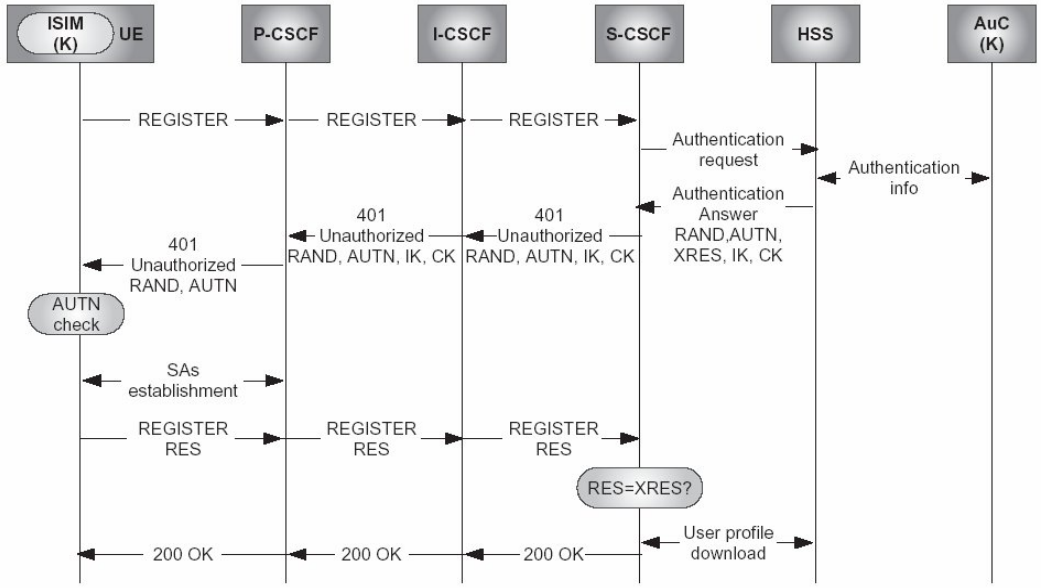

Figure 3.3.1-4: Enregistrement IMS avec authentification AKA

La Figure 3.3.1-4 présente le mécanisme d'AKA :

- L'application ISIM et le réseau partagent un secret à long terme (K).
- L'algorithme AKA est intégré sur un dispositif de puce appelée Universal Integrated Circuit Card (UICC).
- Le réseau produit un vecteur d'authentification basé sur K et un numéro de séquence SQN. Le vecteur d'authentification contient:
    - Un défi aléatoire (RAND)
    - un jeton d'authentification (AUTN)
    - La réponse d'authentification attendue (XRES)
    - Les deux clés de session: une clé d'intégrité (IK) et une clé de chiffrement (CK)

- Le vecteur d'authentification est téléchargé à partir du HSS à la S-CSCF lorsque le premier message Register est reçu par le S-CSCF.
- Le S-CSCF crée une demande d'authentification contenant RAND, AUTN, IK, et CK, et l'envoie à l'utilisateur dans le SIP 401 (Unauthorized).
- L'utilisateur vérifie avec l'ISIM que l'AUTN est correcte. Si l'AUTN est correcte, le réseau a été authentifié. L'utilisateur produit alors une réponse d'authentification en utilisant K et RAND, et envoie le résultat (RES) à la S-CSCF dans un message REGISTRE seconde.
- Le S-CSCF compare RES avec XRES. Si elles correspondent, l'authentification est réussie.

Après ces étapes, deux clés de session sont obtenues (IK et CK). L'équipement utilisateur et le P-CSCF utilisent ces clés de session pour sécuriser la communication par la création de deux paires de canaux d'IPSec, à travers lesquelles le trafic est transmis sous forme cryptée et l'intégrité protégée. Une fois que les SA (Security Associations) sont établies, le P-CSCF permettra d'identifier toutes les requêtes SIP à venir au travers de l'association SA comme appartenant à l'utilisateur authentifié [NiAT02].

3.3.2. Aspect securité de P2P SIP centralisé

Dans le cadre de la sécurité P2P SIP centralisée, plusieurs mécanismes de sécurité peuvent être appliqués. Ceux-ci dépendent du niveau de sécurité requis que nous définissons en trois degrés.
Le premier niveau est le plus simple qui suppose que tous les pairs se font confiance. A partir du moment où le pair fait partie du réseau il est sûr (un mécanisme d'accès préalable peut être effectué, tel une connexion à la box sécurisée par mot de passe.) Le mécanisme met en place un contact direct comme indiqué dans Figure 3.3.2-1.

Figure 3.3.2-1: Confiance mutuelle

Dans la deuxième solution, où les utilisateurs ne se font pas confiance, nous pouvons utiliser la solution de la clé pré-partagée avec le serveur d'authentification pour négocier la génération de clé de session en envoyant la demande de ticket crypté par clé pré-partagée entre les deux utilisateurs (Bob et Alice). L'échange de la clé entre le serveur d'authentification et chaque utilisateur se fait au début du déploiement. Si sa demande de ticket est vérifiée, le serveur d'authentification répond un ticket qui est également chiffré avec la clé pré-partagée, comme indiqué dans la Figure 3.3.2-2. Ils peuvent utiliser ce ticket pour générer une clé de session avec un temps d'expiration. Il s'agit d'un concept similaire à celui proposé par Kerberos.

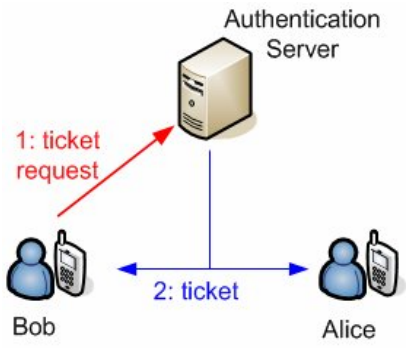

Figure 3.3.2-2: Génération d'une clé de session

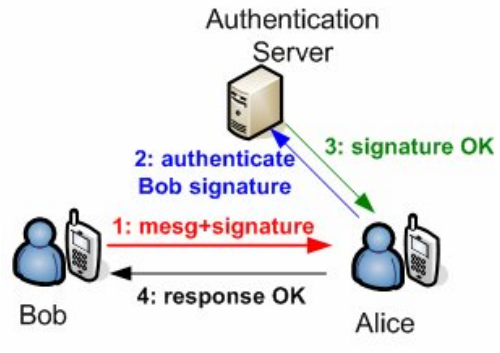

Figure 3.3.2-3: Utilisation d'une clé pré-partagée

Le troisième mécanisme est mis en œuvre à l'aide de clé pré-partagée entre le serveur et l'utilisateur comme le montre la Figure 3.3.2-3. Bob envoie le message qui est signé par sa clé pré-partagée à Alice. Alice vérifie la signature du message en l'envoyant au serveur d'authentification. Alice répond à Bob, si la signature du message est vérifiée sinon si la vérification a échoué elle laisse tomber son message.

En fonction du niveau d'efficacité de la protection souhaité, de nombreux types d'architectures de sécurité peuvent être intégrés. Des mécanismes peuvent être définis tels qu'un serveur AAA centralisé, un serveur pour générer des tickets de session pour les clients, l'authentification par serveur de proximité les signatures par les pairs et la cryptographie par clés public/privé. Il pourrait également être utilisé un mécanisme par protocole défi/réponse protocole d'authentification mutuelle. Cependant, l'augmentation du niveau de protection peut conduire à diminuer les performances du système. Cela devrait être pris en considération avant de décider d'appliquer un mécanisme de sécurité. C'est pourquoi nous nous sommes attachés dans ce travail à évaluer le coût des architectures.

Afin d'évaluer précisément le coût d'une solution nous avons dans une première démarche expérimenté les solutions proposées.

3.4. Expérimentations

Dans un premier temps nous évaluons les produits utilisés pour réaliser notre architecture, sachant que nous utilisons des produits libres. Nous explorons les vulnérabilités Open IMS. Dans un second temps nous avons mis en oeuvre une gestion de service SIP P2P pour comparer les performances des deux solutions.

3.4.1. Expérimentions Open IMS

1) Environnement de test

L'Open IMS Core [MAGE] est une implémentation Open Source d'IMS Call Session Control Functions (CSCFs) et d'un Home Subscriber Server (HSS), qui forment ensemble les éléments de base de toutes les architectures IMS / NGN (comme indiqué aujourd'hui au sein de 3GPP, 3GPP2, ETSI TISPAN et PacketCable l'Initiative). Ces quatre composants sont tous basés sur des logiciels Open Source (par exemple, le SIP Express Router (SER) et MySQL) (Figure 3.4.1-1).

Nous présentons à la suite les tests de sécurité sur la plateforme en vue de nous guider dans la définition d'un niveau de sécurité. Nous testons certaines menaces de sécurité par exemple, les attaques de déni de service (DoS: Denial of Service) et d'usurpation.

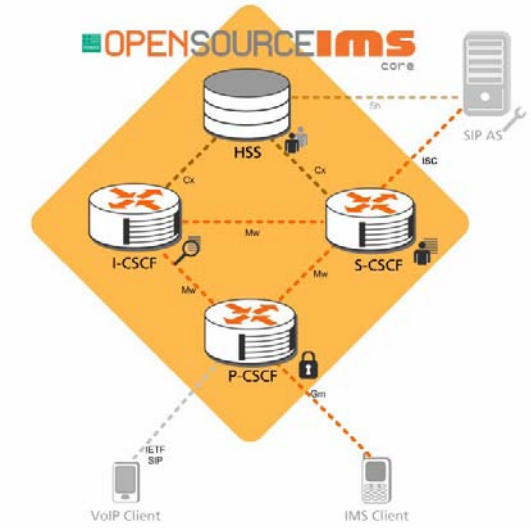

Figure 3.4.1-1: L'architecture d'Open IMS

Notons que OpenIMS est encore en développement et vise à être le logiciel de référence pour tester l'architecture IMS. Il ne fournit cependant pas le même niveau de sécurité que celui proposé dans [ETSI11] par l'ETSI TS 187 003.

2) Expérimentation d'attaques de sécurité

Nous indiquons que l'environnement de test utilise *Wireshark* pour renifler les données sur les ports 4060 5060 et 6060 pour le trafic SIP et aussi surveiller les ports 3868, 3869 et 3870 pour le trafic *Diameter*.

- **Identifier OpenIMS serveur et les informations client**

La première étape consiste à rechercher l'empreinte digitale. Voici une commande pour recueillir les informations du serveur. Nous utilisons des scripts python [Gauc] qui analysent la réponse du serveur OpenIMS qui fonctionne normalement sur le port 5060. Toutefois, si le port n'est pas connu, nous pouvons demander *svmap* (à partir de nmap) dans la recherche.

OpenIMSCore Server:

```
$ python svmap.py –p5060-5062 147.127.240.91 –m INVITE –v
      INFO:DrinkOrSip: 147.127.240.91:5060 -> 147.127.240.91:5060
               -> SIP EXpress router (2.1.0-dev1 OpenIMSCore (i386/linux))
      SIP EXpress router (0.9.6 (i386/linux))
      SIP Device : 147.127.240.91:5060
      User Agent : SIP EXpress router (2.1.0-dev1 OpenIMSCore (i386/linux))
      Fingerprint : SIP EXpress router (0.9.6 (i386/linux))
```

Figure 3.4.1-2: Utilisation de *svmap* pour trouver l'empreinte digitale du serveur

La Figure 3.4.1-2 indique que SIP Express router (serveur OpenIMS base) est exécuté sur le port 5060 sur la machine IP 147.127.240.91.

OpenIC Lite Client:

```
$ python svmap.py –p5060-5062 147.127.240.91 –m INVITE –v
    INFO:DrinkOrSip: 147.127.240.91:5061 -> 147.127.240.91:5061
            -> Fraunhofer  FOKUS/NGNI Java IMS UserEndpoint FoJIE 0.1
(jdk1.3)
            -> T-Com Speedport W500V / Firmware v1.37 MxSF/v3.2.6.26
    SIP Device : 147.127.240.91:5061
    User Agent : Fraunhofer FOKUS/NGNI Java IMS UserEndpoint FoJIE 0.1
(jdk1.3)
    Fingerprint : T-Com Speedport W500V / Firmware v1.37 MxSF/v3.2.26
```

Figure 3.4.1-3: Utilisation de *svmap* pour trouver l'empreinte digitale du client

La Figure 3.4.1-3 montre que OpenIC Lite (OpenIMS client) est exécuté sur le port 5061 sur la machine IP 147.127.240.91.

Une fois l'enregistrement effectué, nous pouvons utiliser cette information, afin de trouver des vulnérabilités. Les attaques suivantes peuvent être ajoutées dans les bases de données de vulnérabilités :

- Symantec's SecurityFocus
- Open Source Vulnerability Database
- National Vulnerability Database

A partir de là, il est possible de lancer des attaques sur OpenIMS.

Menaces de sécurité OpenIMS

Nous avons testé 2 menaces de sécurité. Ce sont le DoS et l'usurpation de l'identité.

3.4.1.1. Les attaques de déni de services avec OpenIMS

Le déni de services (DoS) [HaRe06] inonde le réseau avec des messages dans le but de rendre les utilisateurs indisponibles. Nous allons inonder le serveur OpenIMS par des messages SIP INVITE. Nous testons en utilisant le programme inviteflood qui est écrit en langage C à partir de [CoOb].

#./inviteflood eth0 bob open-ims.test 147.127.240.91 10000

Cette commande va envoyer 10,000 SIP INVITE demandes à alice@open-ims.test. En conséquence, l'utilisateur Bob devient indisponible, car il ne peut pas accepter l'appel de personne, sauf gérer cette inondation (Figure 3.4.1-4).

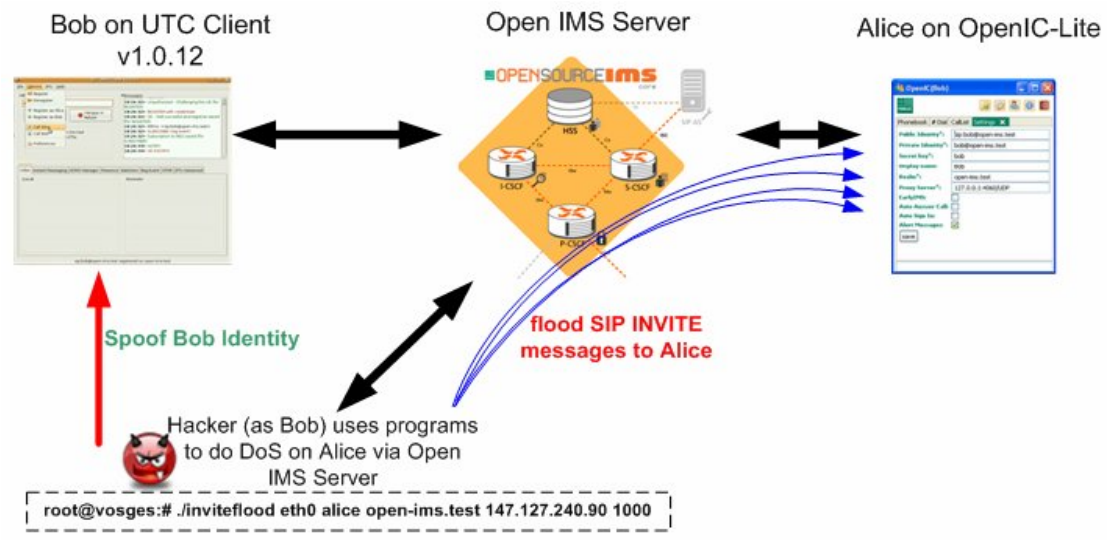

Figure 3.4.1-4: Attaque par inondation du serveur OpenIMS

3.4.1.2. L'usurpation d'identité dans OpenIMS

L'usurpation d'identité consiste à prendre l'identité d'un autre utilisateur pour demander des informations en vue de commettre un acte malveillant ou criminel. Nous constatons que cette attaque peut facilement être réalisée. Par exemple, la commande suivante permet à un utilisateur de prendre l'identité d'Alice auprès de Bob.

#./inviteflood eth0 bob open-ims.test 147.127.240.91 –a alice

Puis, Bob croit que l'appelant est la vraie Alice.

3.4.1.3. Synthèse : OpenIMS et les solutions NGN

OpenIMS system est intégré à tous les composants IMS (par exemple, P/I/S-CSCF, HSS). Il est très populaire pour de nombreux développeurs et chercheurs de la communauté IMS, car il est facile d'utilisation. Toutefois, il est à préciser que OpenIMS n'est pas destiné à devenir ou à agir comme un produit dans un contexte

commercial [MAGE]. Son unique objectif est de fournir une implémentation de référence de base pour des outils de tests IMS et de développements d'applications.

Cependant dans la mesure où la technologie IMS peut faire l'objet de brevets, que son coût de licence est important, il peut s'avérer intéressant d'utiliser un source libre à condition comme nous le relevons de rajouter les modules de sécurité manquants [ETSI11].

3.4.2. Expérimentations P2P SIP Centralisée

Il n'y a à notre connaissance pas de source libre de P2P SIP centralisée pour la prestation de services à domicile. Nous avons donc développé l'environnement de test pour le service de partage de photos en utilisant Java et la bibliothèque JAIN-SIP [RaBe]. L'application déployée se compose de 3 phases: la publication, la recherche et la récupération.

Les principales composantes du service (Figure 3.4.2-1) sont les suivants:

- Global/ Local Manager: gère la liste de contacts pour le serveur / client
- Contact List: contient la liste d'adresse des amis, des données et des index (adresse des ressources)
- SIP UA (User Agent) : SIP UA est utilisé pour être des interfaces SIP à d'autres utilisateurs. Il est utilisé pour établir la session pour les applications spécifiées.

Un nœud stocke ses données et les index. Les nœuds publient leurs index de partage au gestionnaire de catalogue. Le gestionnaire de catalogue (Catalogue Manager) maintient les index de partage. La liste de contact (Contact List) stocke une liste d'amis par nœuds, qui est gérée par le gestionnaire local.

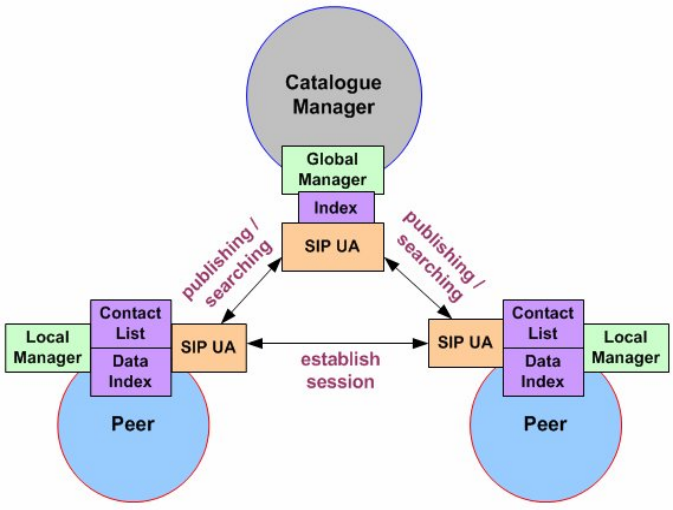

Figure 3.4.2-1: Principaux composants du service du service de références

La Figure 3.4.2-2 montre une capture d'écran obtenue par un traçage *Wireshark* lors de l'exécution du service. Cette application est basée sur une signalisation SIP pour

établir une communication entre utilisateurs. Le service est défini en attachant des éléments XML avec le corps du message SIP (type MIME) [FrBo96] dans le format text/plain. Par exemple, le message Publishing contient en attaché l'index de la photo à publier. En outre, le service de protocole de communication est mis en correspondance avec le POJO (Plain Old Java Object : une classe java qui contient uniquement les membres de la classe avec les méthodes associées) pour faciliter la gestion. Les clients établissent des sessions SIP avec le serveur de catalogue et peuvent directement partager entre eux des photos.

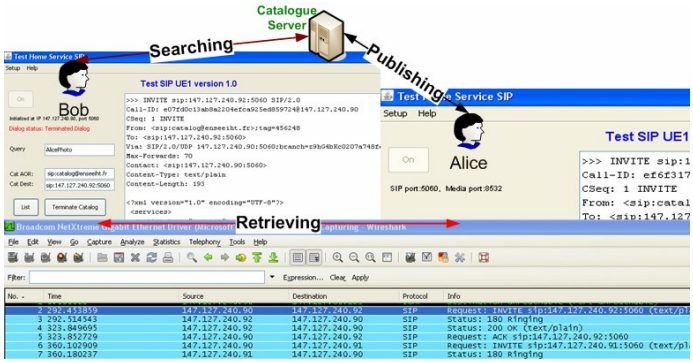

Figure 3.4.2-2: Capture d'écran du service de partage des photos

3.5. Analyse et comparaison des mécanismes de sécurité

L'étude se concentre sur la signalisation induite dans l'architecture par l'enregistrement, la publication, la recherche et la récupération, avec différents mécanismes de sécurité.

3.5.1. Description de la signalisation

Nous proposons 3 mécanismes de sécurité pour l'architecture P2P centralisée avec le protocole SIP. Ces trois mécanismes sont représentatifs de différents niveaux de sécurité (faible moyenne forte) et d'architectures (intégrée, séparée). L'architecture intégrée regroupe les fonctions de gestion de données et d'authentification sur une seule machine alors que l'architecture séparée, comme son nom l'indique les sépare (comme dans IMS) sur des machines différentes. La première est plus simple en termes d'échange mais supporte plus de charge sur une même machine que la seconde.
Nous détaillons à la suite les clients SIP qui se distinguent par leurs mécanismes d'authentification. Plus précisément, un client qui souhaite accéder à un réseau IMS utilise obligatoirement IPSec pour se connecter au proxy. Quant aux clients SIP1,2 et 3, ils proposent une sécurité légère (SIP Client 1) jusqu'à une sécurité renforcée (SIP Client 3).

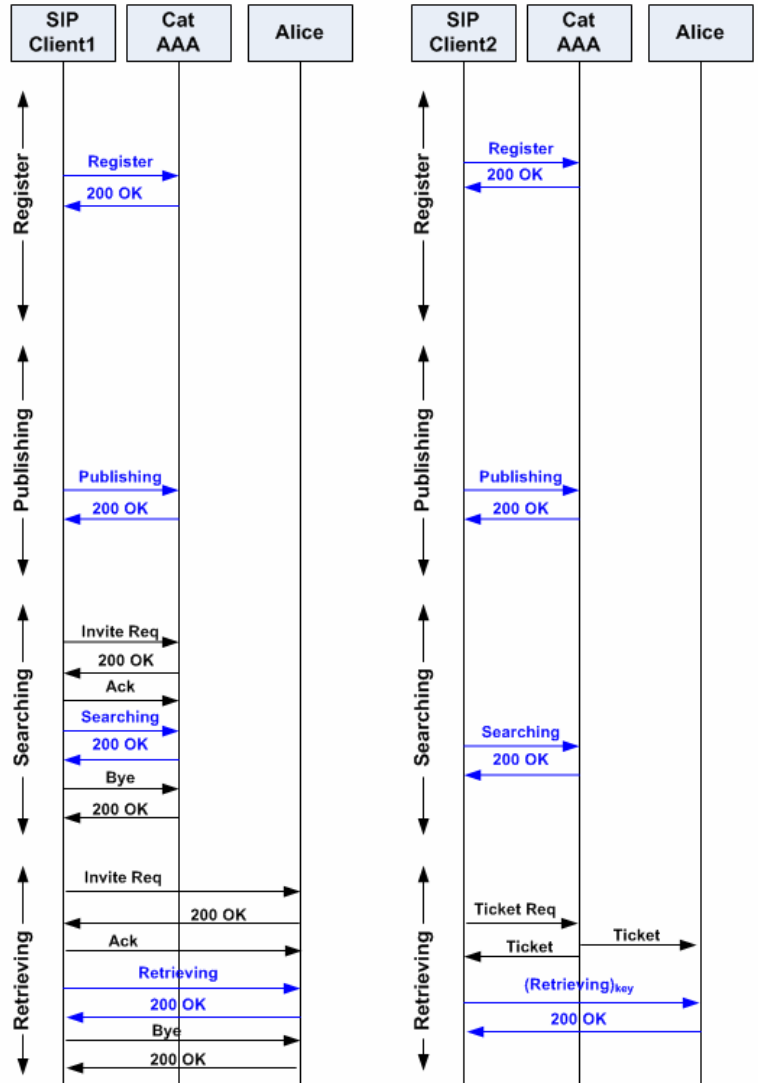

Figure 3.5.1-1: Signalisation d'un client SIP1 et d'un client SIP2

Le premier mécanisme, SIP client1 (Figure 3.5.1-1), repose sur une signalisation SIP standard sans contrôle entre les utilisateurs (mécanisme de niveau 1). Les deux autres SIP client 2 (Figure 3.5.1-1) et SIP client 3 (Figure 3.5.1-2) utilisent un SIP modifié pour appliquer la sécurité (changement de messages).

Le catalogue et le serveur d'authentification sont intégrés sur la même machine pour tous les services SIP client exceptés pour la solution nommée SIP client3 où le catalogue et le serveur AAA sont séparés.

- SIP client 1 utilise le protocole de base SIP, il s'enregistre auprès du serveur d'authentification (AAA). Durant l'étape de Publication, les adresses des ressources partagées sont ajoutées au message SIP PUBLISH standard qui est envoyé par le client. Au début de la phase de recherche, le SIP client1 initie une ouverture de session en envoyant un message SIP INVITE avant d'envoyer ses mots clés. Ensuite, il peut envoyer des mots clés spécifiques pour rechercher la ressource qu'il désire. Après avoir trouvé une ressource, le

nœud entame la phase de récupération qui consiste à se connecter directement au domicile du propriétaire (e.g., Alice), ensuite le client termine la session. Notons que SIP client 1 peut-être attaqué durant l'étape de récupération car il n'y a aucune authentification. Pour éviter ces problèmes de sécurité nous proposons de mettre en place des mécanismes de sécurité entre les utilisateurs SIP2, ou SIP3

- SIP client 2 propose de créer une clé de session à partir d'un serveur d'authentification en envoyant la demande de ticket et en attendant un ticket de réponse. Ce ticket est chiffré par la clé pré-partagée de chaque utilisateur. Après cela, les utilisateurs (SIP client 2 et Alice) ont le ticket, il est utilisé pour créer une clé de session. SIP client 2 utilise la clé de session générée pour communiquer avec Alice. Pour accélérer le processus, les mots clés relatifs à la recherche (ex poisson rouge) sont ajoutés au contenu du message SIP (SIP Option) émis durant la phase de recherche. On obtient alors un processus plus rapide que SIP client 1 qui utilisant le SIP standard, nécessite une ouverture de session (en utilisant les messages SIP INVITE/TRYING/OK/ACK) pour pouvoir émettre des données spécifiques au service, en l'occurrence les mots clés.
- SIP client 3 propose de séparer le catalogue et le serveur d'authentification. Dans cette solution, le serveur d'authentification vérifie la signature de chaque utilisateur à l'aide d'une clé partagée avec ce dernier. Dans l'étape d'enregistrement, le SIP client 3 envoie le message Register et sa signature en utilisant HTTP Digest avec la clé pre-partagée au catalogue. Le catalogue transmet au serveur l'authentification pour vérifier la signature de SIP client 3. Si la signature est correcte, le serveur envoie le message 200 OK au catalogue. Le catalogue transmet le message 200 OK au SIP client 3 avec sa signature. SIP client 3 envoie ce message au serveur d'authentification pour vérifier la signature du catalogue. Si la signature est correcte, le serveur envoie le message 200 OK au SIP client 3. Dans l'étape de rapatriement, Retrieving, la signalisation est similaire à celle de l'étape Register comme indiqué dans la Figure 3.5.1-2. Par ailleurs, les échanges d'authentifications des clients à la demande des clients de la phase Retrieving peut permettre d'obtenir une clé de session qui sera utilisée pour la communication directe entre clients.

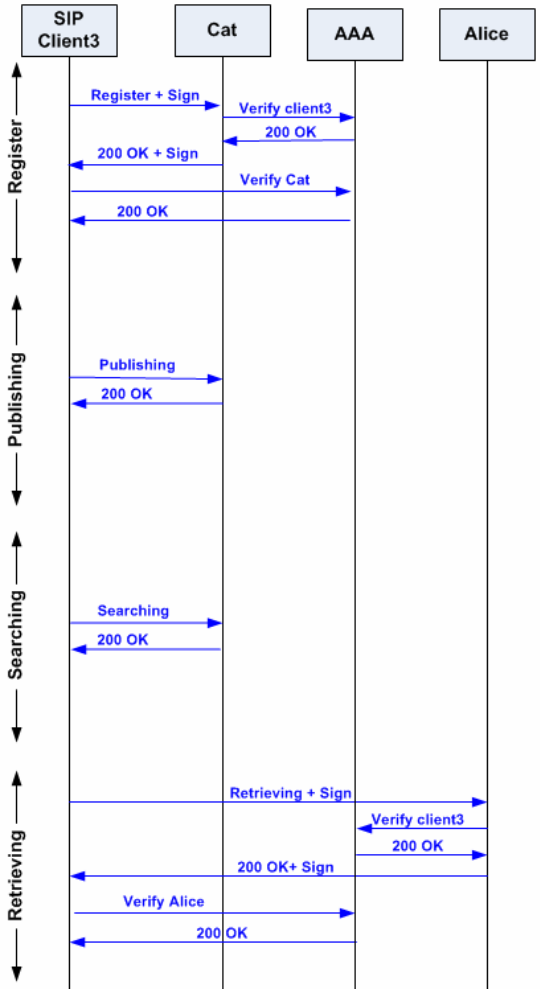

Figure 3.5.1-2: Signalisation SIP Client 3

Une synthèse de la signalisation des différentes solutions IMS, SIP1, 2 ou 3 côtés client est présentée dans la Figure 3.5.1-3. La solution SIP1 induit des sessions avec le catalogue pour la publication et la recherche, et une session (en utilisant SIP INVITE) avec le client partageur pour récupérer son contenu. La signalisation du client est alors semblable à celle de l'architecture IMS à l'exception de la partie authentification. En ce qui concerne la signalisation de SIP 2, et SIP 3, elle est réduite par l'utilisation d'un SIP avec options qui permet de faire passer des données spécifiques au service.

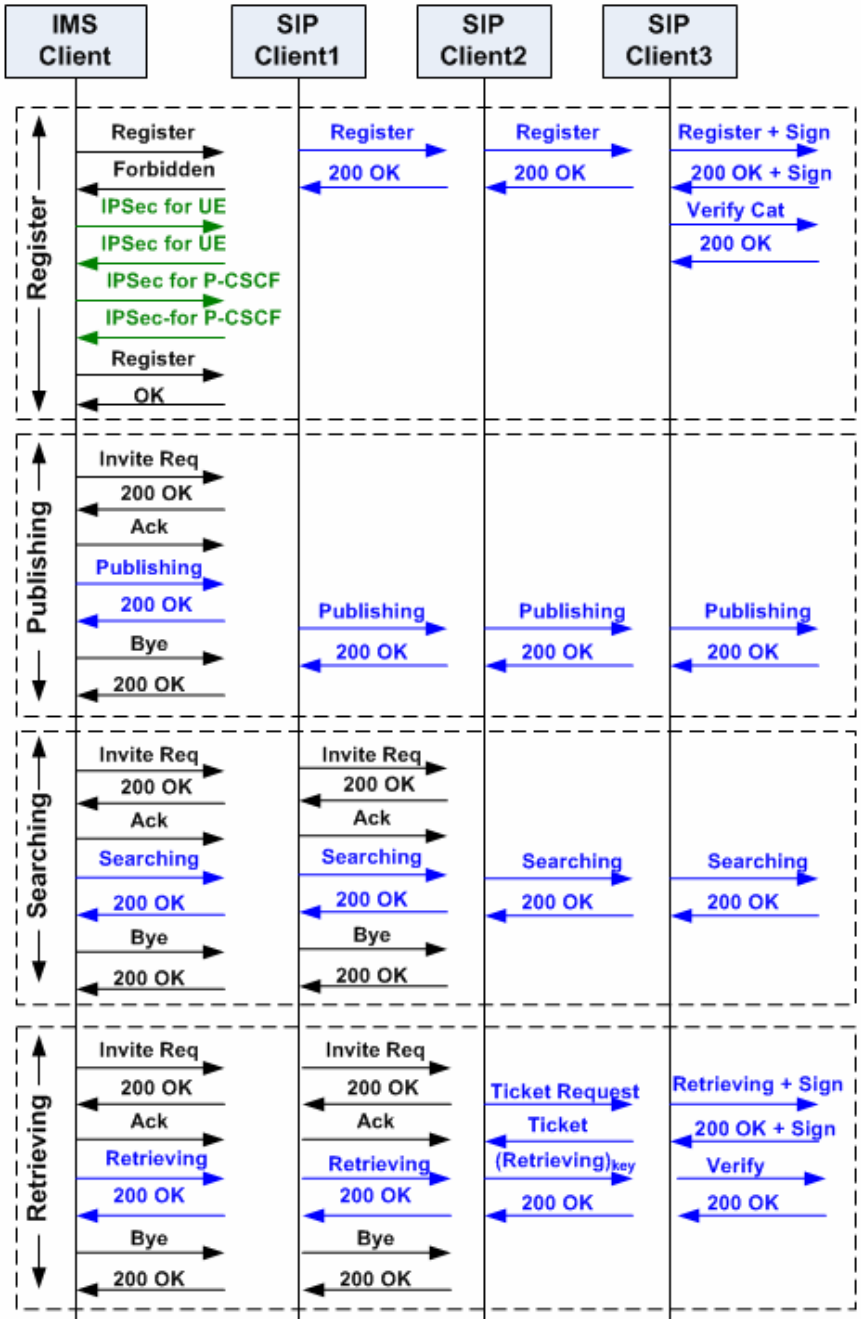

Figure 3.5.1-3: Signalisation côté client

Notons que la signalisation nécessaire au fonctionnement du service de partage de photos est de 2 échanges : requête/réponse, et qu'il est possible de l'intégrer aux requêtes SIP. Cette technique d'inclusion peut être étendue à tout type de services requête/réponse basé sur le même type de signalisation.

3.5.2. Comparaison des signalisations client

La signalisation est analysée du point de vue de l'utilisateur: il s'agit de compter le coût de la procédure en nombre de messages (par exemple, l'envoi et la réception par

l'UE) et la taille de ces messages. Les messages étant de même type, nous considérons que les compter est suffisant.

Analysons la signalisation de l'IMS et des architectures SIP centralisées.

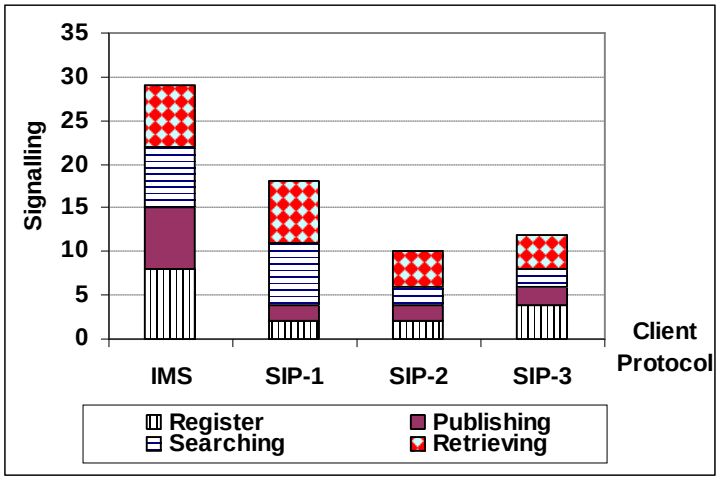

Figure 3.5.2-1: Poids de la signalisation côté client

La Figure 3.5.2-1 présente la signalisation échangée durant chaque phase et leur total côté client. Nous pouvons voir que le client IMS génère plus d'échange que les clients SIP. Cependant, la solution IMS propose une solution de sécurisation de niveau très élevé, mais si on lui retire le mécanisme IPSec on obtiendrait une sécurité de niveau élevé type SIP 3 et donc moins coûteuse. Il faut également noter que la signalisation IMS peut également être diminuée en utilisant un SIP non standard avec passage d'information de service dans les messages SIP.

3.5.3. Comparaison des signalisations réseau

La première évaluation donne un point de vue global sur le côté client, mais il ne révèle pas la complexité de toute l'architecture. Le deuxième point est d'évaluer le surcoût de la signalisation sur la totalité du réseau. Une vision du comportement global est obtenue en comptant les messages échangés entre tous les nœuds.

La Figure 3.5.3-1 montre la signalisation globale échangée sur la totalité du réseau pour les différentes solutions. Elle indique la signalisation induite par un client qui souhaite obtenir une ressource auprès d'un autre client.

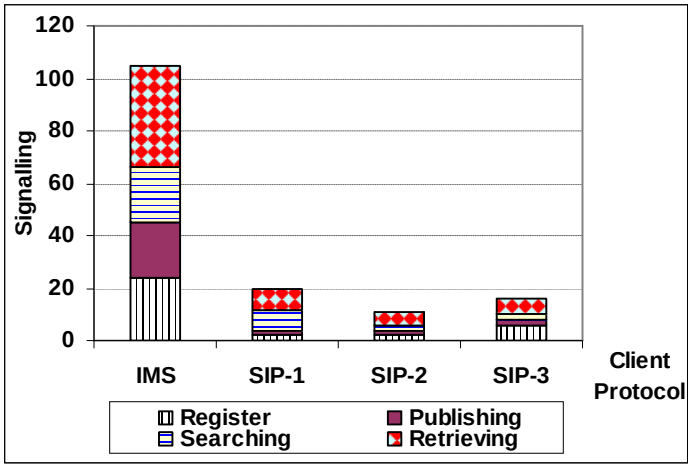

Figure 3.5.3-1: Poids de la signalisation globale dans le réseau pour une recherche

IMS génère une signalisation importante car sa norme exige plusieurs composants (par exemple, CSCF, HSS et ASs) pour créer et établir la session. En revanche, avec une architecture SIP centralisée il y a seulement un composant de gestion auquel est adjoint différents niveaux de sécurité : 1) l'authentification par mot de passe (SIP-1): la connexion est établie une seule fois sur le réseau, 2) l'authentification par ticket Kerberos (SIP-2): le serveur Kerberos émet un ticket pour les communications entre nœuds, et l'authentification des signatures 3) (SIP-3): les nœuds ont à signer toute la signalisation et il y a vérification à chaque fois auprès du serveur d'authentification.

L'importance de la signalisation IMS en regard des solutions SIP est particulièrement marquée durant la phase d'enregistrement (Register) qui a lieu à la connexion du système par le client, ou lors d'un déplacement de celui-ci. Il peut être supposé que cette phase a lieu somme toute moins fréquemment que les autres phases. De même la phase de publication peut être considérée comme peu fréquente en regard de l'occurrence de celle de recherche.

La Figure 3.5.3-2 montre un résumé de la signalisation générée par cinq phases de recherches et de récupération sans prendre en compte l'enregistrement. Nous pouvons voir que SIP-3 induit encore moins de signalisation que la norme IMS, mais que le ratio diminue.

Si le système est très utilisé une solution comme IMS qui propose une très grande sécurité sera relativement moins coûteuse que pour un service utilisé peu fréquemment ; quant aux solutions SIP 2 et SIP3, le coût de signalisation va être globalement similaire sur le réseau lorsque l'utilisation du service croît.

Un bon compromis est alors d'utiliser la solution SIP-3. Elle permet d'avoir un bon niveau de sécurité avec un coût de signalisation faible (relativement aux autres solutions)

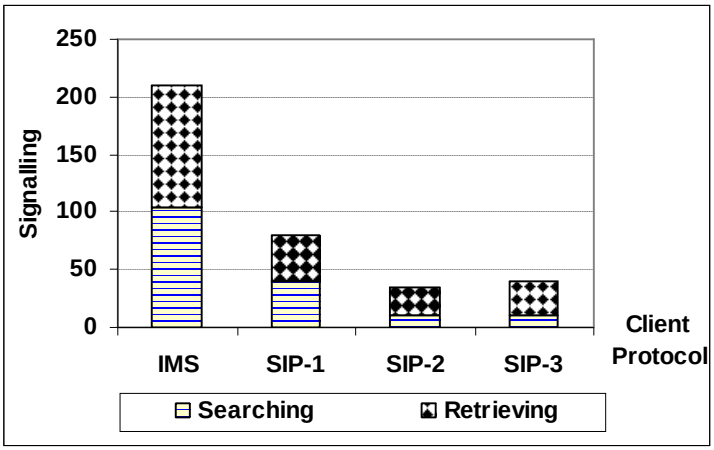

Figure 3.5.3-2: Poids de la signalisation pour cinq recherches

3.6. Choix d'architecture : solution SIP par proxy

Nous avons présenté 3 solutions de pair à pair centralisées [WeFB10] qui offrent l'inconvénient d'un risque de déni de service en cas d'inadéquation entre l'offre en capacité du point central et la demande des utilisateurs. Ce risque devient plus important lorsque la taille du réseau P2P augmente. C'est pourquoi nous proposons une architecture avec des éléments de proximité (nommés SIP proxy) assez similaire à celle de IMS. Avec l'aide des proxy SIP, les utilisateurs peuvent définir certains filtrages de sécurité et répartir des tâches de vérification à partir d'un serveur centralisé, la sécurité est ainsi augmentée car les risques d'attaques peuvent être protégés par un proxy.

Par ailleurs, nous proposons, suite à l'analyse précédente, d'utiliser les mécanismes de sécurité de SIP3 à savoir HTTP Digest [FBHL99] avec la clé pré-partagée entre les utilisateurs et le serveur d'authentification (AAA).

3.6.1. Fonctionnement SIP proxy et HTTP digest

Les étapes principales de cette authentification sont les suivantes:

Amorçage (bootstrap):

L'adresse du proxy est manuellement pré-installée par un opérateur de réseau. Quand un nœud rejoint le réseau, il doit s'enregistrer auprès de son proxy. L'illustration de cette solution est montrée sur la Figure 3.6.1-1. Elle requiert d'avoir des clés pré-partagées entre tous les utilisateurs et le serveur d'authentification. Nous supposons que les liens entre les proxy et le serveur d'authentification sont sécurisés. Cette hypothèse est raisonnable puisque les proxy sont habituellement fixes.

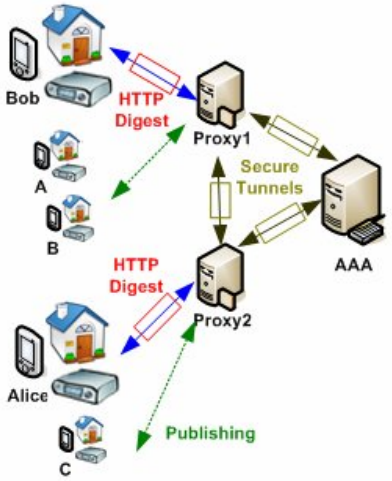

Figure 3.6.1-1: Authentification dans la solution SIP proxy

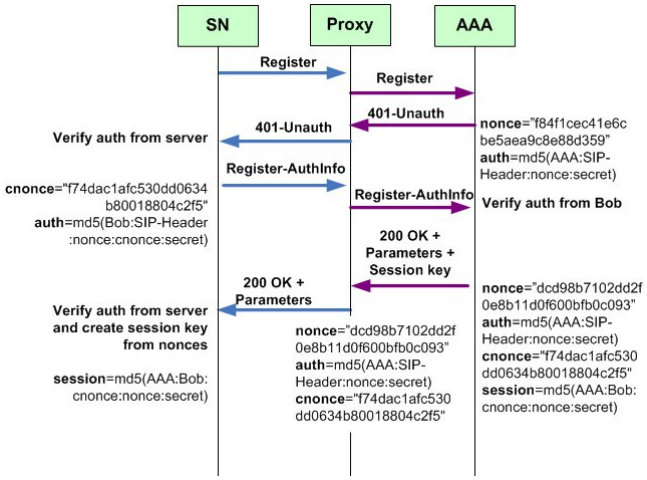

Figure 3.6.1-2: Utilisation de HTTP Digest pour l'enregistrement SIP

Enregistrement: Toutes les signalisations pour établir les sessions se font via le proxy. Dans la Figure 3.6.1-2, SN envoie le *SIP Register* au serveur d'authentification (via le proxy). Le serveur d'authentification retourne le message *401-Unauthentication* à SN (nœud source) avec ses paramètres (ex, nonce, auth). SN utilise sa clé pré-partagée et les paramètres précédents (nonce) à partir du serveur d'authentification pour générer des informations d'authentification, et retourne le message *SIP Register* avec le champ *Authentication Info* au serveur d'authentification. Ensuite, le serveur d'authentification vérifie les informations d'authentification retournées. Si, elles sont correctes, le serveur d'authentification renvoie *SIP 200 OK* avec les paramètres pour une génération de la clé session et également une clé session car le proxy n'a pas la clé pré-partagée. Le proxy supprime la clé de session contenue dans le message reçu et le transmet : *SIP 200 OK* avec les paramètres pour créer la clé session pour le lien entre SN et le proxy qui n'est pas sécurisé. Enfin, SN génère la clé de session à partir des paramètres reçus avec la clé pré-partagée et démarre la création d'un tunnel sécurisé vers le proxy en utilisant cette clé de session.

Dans ce processus, il y a 4 messages échangés entre SN et Proxy, et 4 messages échangés entre le proxy et le serveur d'authentification.

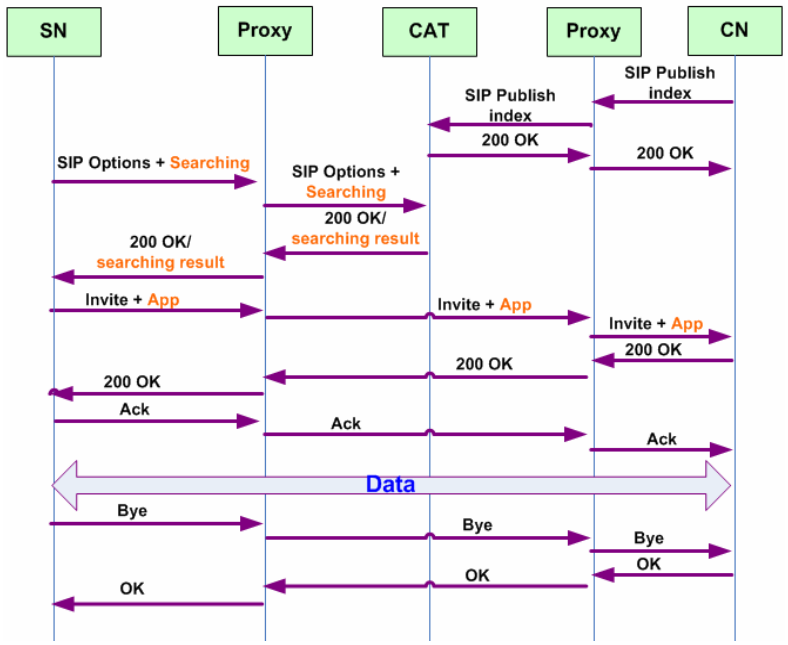

Figure 3.6.1-3: Diagramme de signalisation en architecture SIP proxy

La Figure 3.6.1-3 décrit un scénario en 3 phases : la publication, la recherche et la récupération d'une ressource.

Publication: Cette étape consiste à publier l'index de la(es) ressource(s). Elle est réalisée par un client CN (nœud correspondant) qui partage des ressources. CN publie les index au serveur de catalogue (CAT) en passant par son serveur proxy. Les index sont joints au message *SIP PUBLISH*. Il n'y a pas besoin d'émettre au préalable un *SIP INVITE* comme c'est le cas dans IMS car nous avons allégé les messages. Le serveur de catalogue répond par un message *SIP 200 OK* pour indiquer que la publication est réussie.

Il y a 2 messages entre CN et le proxy et 2 messages entre le proxy et le serveur de catalogue.

Recherche: SN envoie la requête *SIP OPTIONS* au serveur de catalogue à travers le proxy, il n'y a pas de SIP INVITE comme c'est le cas dans IMS. *SIP OPTIONS* est une requête de demande d'index et qui contient des mots clés. Le serveur de catalogue trait la requête et retourne l'index de la ressource dans le message *SIP 200 OK*.

2 messages sont échangés entre SN et le proxy et 2 messages entre le proxy et le serveur de catalogue.

Récupération: SN récupère (télécharge) la ressource à partir de l'index. Pour cela, SN ouvre une session avec CN (propriétaire de la ressource) en envoyant la requête

SIP INVITE contenant des paramètres spécifiques à l'application (ici le service de photo avec comme paramètre *photo URI*). Cette session passe par son proxy et le proxy du CN. CN accepte la requête en envoyant *SIP 200 OK* à SN. SN renvoie *SIP ACK* à CN. Puis, SN télécharge la ressource directement (sans passer par le proxy) et termine la session en envoyant *SIP BYE* et en recevant la *SIP OK*. 5 messages sont échangés.

3.6.2. Preuve de concept de l'architecture SIP proxy

Nous avons effectué une preuve de concept de notre architecture en créant un réseau de test expérimental avec plusieurs ordinateurs qui met en œuvre les éléments : SIP client, SIP proxy, le serveur de catalogue et le serveur d'authentification, comme indiqué dans la Figure 3.6.2-1.

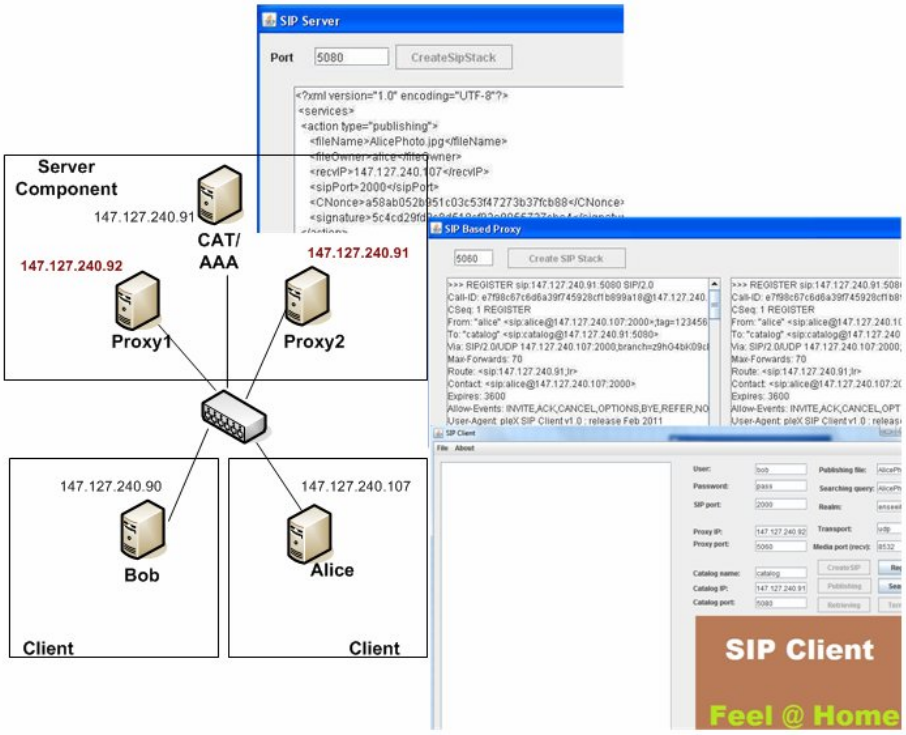

Figure 3.6.2-1: Banc de test de la solution SIP proxy

Nous séparons les composants en composants orientés serveur et client. Le côté serveur contient les éléments serveur de catalogue, serveur d'authentification et proxy. Le côté client contient les éléments SIP clients (par exemple, Bob, Alice).

Le scénario de la Figure 3.6.2-1 inclue 3 composants et 2 réseaux (voir la Figure 3.6.1-1). Notre réseau expérimental est constitué de 4 ordinateurs et d'un switch Ethernet. Nous avons configuré tous les composants dans le même réseau. Par ailleurs le serveur de catalogue et le serveur d'authentification sont intégrés dans le même ordinateur. Cependant les messages sont routés comme dans le scénario décrit dans le diagramme (Figure 3.6.1-3).

La conception de classe

Nous créons 5 composants. Chaque composant contient plusieurs paquetages et chaque paquetage contient plusieurs classes. Le protocole SIP est réutilisé dans chaque composant.

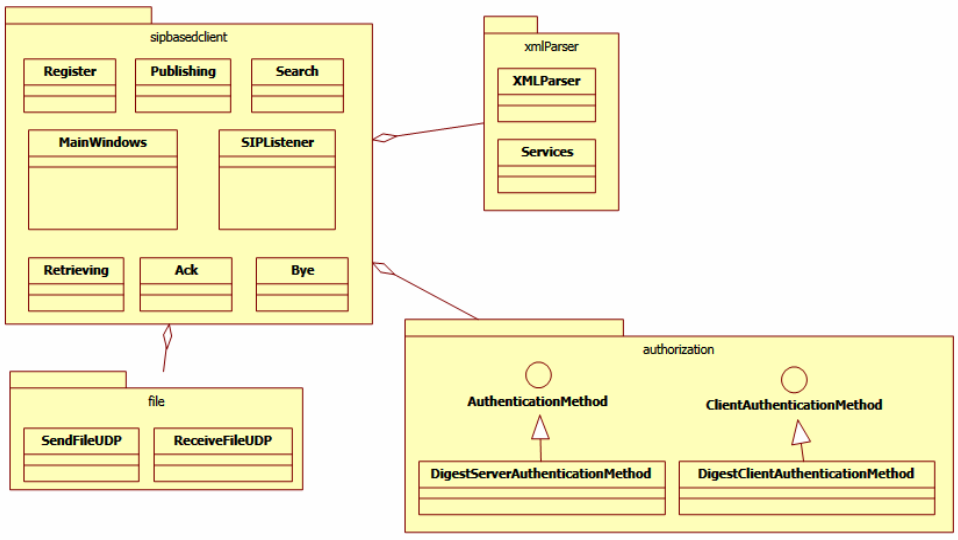

Figure 3.6.2-2: Diagramme des classes pour le client SIP

Comme illustré en Figure 3.6.2-2, le diagramme des classes se compose de 4 paquetages :

Le client SIP:

- Le paquetage sipbasedclient:
  - Il interprète le protocole SIP. *MainWindows* gère une interface graphique du client. *SIPListener* est l'interface de communication avec le message SIP (envoie de demande, réception de réponse). *Register* est utilisé pour se connecter au système. *Publishing* est utilisé pour envoyer une demande de publication de ressources. *Search* est utilisé pour chercher des ressources. *Retrieving*, *Ack* et *Bye* gèrent la signalisation *SIP INVITE* pendant la phase de récupération des données.
- Le paquetage XmlParser:
  - Ce paquetage interprète les éléments XML. XMLParser est utilisé pour définir la fonction de lecture et d'écriture des éléments XML à partir d'objets de services. La classe *Service* est une classe POJO. Elle permet de joindre des éléments XML au message SIP.
- Le paquetage File:
  - Ce paquetage est lié à l'envoi/réception des fichiers sur le réseau. *ReceiveFileUDP* est appelé à partir de *Retrieving* (du paquetage sipbasedclient) quand la récupération de la ressource a commencé. *SendFileUDP* est appelé à partir *SIPListener* lorsque la réponse *ACK* est reçue par un client.

- Le paquet authorization:
    - Ce paquetage gère le mécanisme d'authentification via le protocole HTTP Digest. *DigestServerAuthenticationMethod* définit les méthodes pour calculer tous les paramètres liés à la sécurité (par exemple, nonce, la signature) et vérifier un correspondant pour le serveur. *DigestClientAuthenticationMethod* définit les méthodes pour calculer les paramètres liés à la sécurité et vérifier un correspondant pour un client

Le proxy SIP:

- L'interprétation du protocole SIP dans le proxy SIP est similaire à son interprétation dans le client SIP. Les principaux rôles du proxy SIP sont : accepter (ou non) les entrées dans le réseau, vérifier la signature du client après son enregistrement et maintenir la transaction client et serveur.

Le serveur SIP:

- L'interprétation du protocole SIP dans le serveur SIP est similaire à son interprétation dans le client SIP. Un serveur SIP réalise les rôles d'un serveur de catalogue et d'un serveur d'authentification. Il crée le nonce (nombre aléatoire) à l'étape enregistrement, vérifie les informations d'authentification de la réponse client, crée une clé de session, l'envoie au proxy correspondant et il gère également les index des ressources qui sont publiées par un client.

3.6.3. Evaluation du coût de la solution

Analysons maintenant la signalisation du côté client de l'architecture proxy et comparons la avec celle générée par une sécurité entièrement centralisée, SIP3. Les résultats sont présentés en Figure 3.6.3-1. Les signalisations des deux méthodes sont identiques excepté pour la phrase récupération. Dans cette phase l'élément de proximité ouvre une session en utilisant *SIP INVITE* qui passe par les éléments de proximite du SN et du CN.

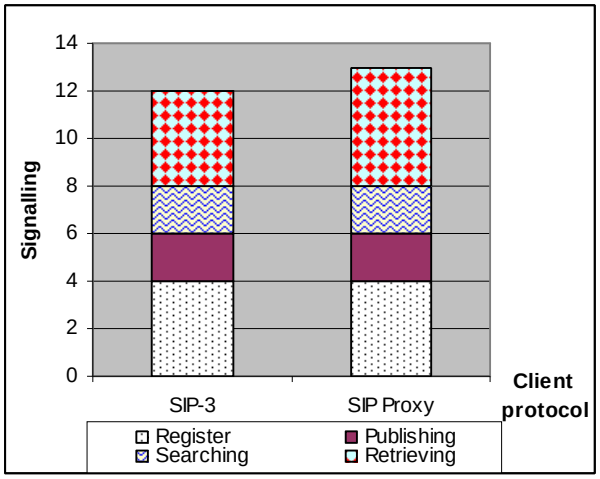

Figure 3.6.3-1: Comparaison des signalisation pour un client SIP3 et un client SIP proxy

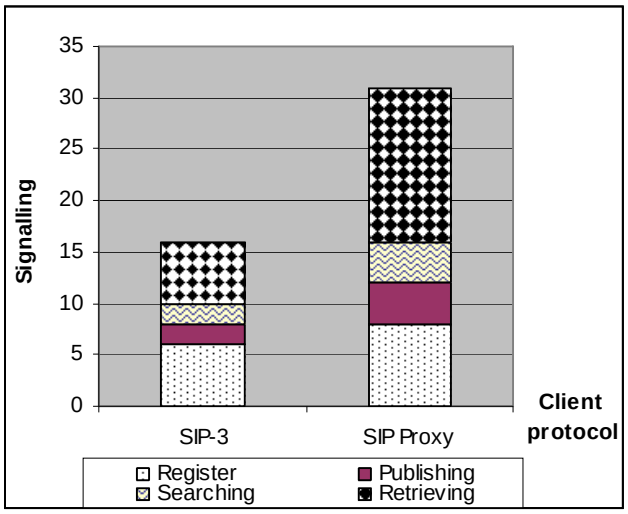

Figure 3.6.3-2: Signalisation globale en SIP3 et en SIP proxy

La Figure 3.6.3-2 montre la signalisation globale. Celle générée par les éléments de proximité augmente le coût de l'architecture. Cependant cette architecture offre certains avantages: (1) un utilisateur n'est pas directement relié au serveur d'authentification ce qui rend plus difficile la compromission du système, (2) il peut être installé sur l'élément proxy une protection système comme un pare-feu ou bien un système de détection d'intrusion qui augmente le niveau de sécurité (3) il utilise conformément à la version SIP standard un message SIP INVITE pour ouvrir une session SIP dans la phase de récupération (4) les tâches du serveur d'authentification sont réduites ; un utilisateur dans SIP-3 doit contacter le serveur d'authentification pour vérifier son correspondant et son correspondant doit également demander une vérification tandis que dans SIP proxy l'authentification des utilisateurs se fait en une seule fois. Les tâches du serveur d'authentification sont déléguées aux proxy après qu'un utilisateur ait été authentifié.

Lorsque le service est davantage utilisé et que le client effectue plusieurs recherches, la signalisation de SIP proxy augmente davantage que celle de SIP-3 puisque SIP proxy ouvre la session en utilisant le message standard *SIP INVITE* ce qui génère plus de coût que dans SIP3. En outre le nombre de composants est plus important que dans SIP-3.

En fait, la solution de SIP proxy est similaire à la solution IMS, le proxy est similaire à l'élément P-CSCF d'IMS. Cependant, elle est simplifiée par l'intégration des composants I/S-CSCF et HSS dans le serveur d'authentification. En outre, le lien entre P-CSCF et l'UE dans la solution IMS utilise IPSec, tandis que notre solution permet l'utilisation d'une clé de session par HTTP Digest, qui est un mécanisme moins lourd, pouvant être plus facilement intégré à des dispositifs de faible capacité, et d'énérgie limitée.

3.7. Conclusion

Dans ce travail, nous avons étudié la sécurisation des architectures de gestion de services reposant sur une architecture centralisée et la possibilité de mettre en oeuvre une architecture SIP alternative à l'architecture standard IMS. L'analyse présentée dans ce chapitre s'appuie sur un service spécifique ; elle est néanmoins adaptée à d'autres types de services en requête/réponse.

Sur la base existante du protocole SIP et l'architecture IMS standard, ce chapitre expose un cadre de sécurité reposant sur le protocole SIP avec des extensions qui est comparé avec la solution IMS. Nous comparons les messages échangés dans les architectures centralisées SIP et IMS en fonction du mécanisme de sécurité utilisé. Trois mécanismes de sécurité sont proposés correspondant à trois niveaux : faible (SIP1), moyen (SIP2) et haut (SIP3); le niveau de sécurité de IMS étant considéré comme très haut.

La solution IMS est la solution qui génère le plus de signalisation, celle-ci peut être réduite par l'utilisation d'un protocole SIP avec extension ainsi que par une diminution du niveau de sécurité qui la rendrait alors comparable à la solution SIP3.

Par ailleurs pour alléger les problèmes de surcharge susceptibles de se produire dans un système centralisé, nous avons proposé d'utiliser une architecture de niveau SIP3 mais avec des éléments de proximité, des proxy, à même de prendre en charge des fonctionnalités du serveur central. Ces équipements jouent un rôle similaire aux éléments P-CSCF mais avec un coût de signalisation moindre. Par ailleurs ils permettent d'augmenter le niveau de sécurité de la solution SIP3 en réalisant des fonctions de filtrage.

De cette analyse nous concluons par l'intérêt d'une solution SIP devant une solution IMS en raison du bon rapport sécurité/coût de signalisation obtenu mais également de critères de réalisation/maintenance. Une solution SIP à partir de logiciels libres est en terme d'achat moins onéreuse et en terme d'évolutivité peut être plus simple à gérer, que la solution très standardisée qu'est IMS.

Chapitre 4

4. Architecture distribuée pour la délivrance de services au domicile

Sommaire

4.1. Introduction

Nous avons présenté la solution P2P centralisée [WeFB10] avec différents mécanismes de sécurité dans le chapitre 4. Cette architecture présente des avantages en termes de gestion et de simplicité. Dans un réseau à grande échelle, cependant, certains désavantages apparaissent : faire confiance à une entité de gestion de ressource, goulot d'étranglement, besoin d'une grande capacité de traitement et de stockage. Bien sûr, d'autres serveurs peuvent être ajoutés afin d'équilibrer la charge, mais cela augmente le coût du prestataire de services en comparaison avec une solution P2P pur. Dans le P2P pur, tous les utilisateurs sont responsables de la gestion des ressources, ce qui augmente la capacité de traitement et de stockage.

Dans ce chapitre, nous présentons l'architecture P2P avec dans un premier temps une étude du modèle de base du pair a pair complètement distribué (pur P2P) puis en considérant une approche en cours de standardisation par IETF : l'approche par DHT de RELOAD. Nous indiquons la conception du logiciel RELOAD que nous avons deployé.

4.2. Architecture P2P pur

Précisons en premier lieu le fonctionnment d'une architecture P2P pur sur le scénario de référence. Nous proposerons ensuite un shema d'authentification que nous évaluerons et analyserons.

4.2.1. Etapes de délivrance de services au domicile

Sur une architecture P2P pur, la prestation de services à domicile se déroule selon les phases de la façon suivante.

Enregistrement: L'architecture est composée des nœuds participants au réseau overlay. Le mécanisme utilisé pour se connecter à un réseau overlay est appelé amorçage (Bootstrapping). Il existe différentes méthodes d'amorçage: (1) télécharger directement une liste de nœuds à partir d'un serveur d'amorçage et établir la connexion avec les nœuds plus proches et (2) écouter les annonces transmises régulièrement en broadcast/multicast par les nœuds du réseau. En outre il est possible qu'un nœud mémorise en cache un nœud de réseau avec qu'il communique pour, lors d'une reconnexion ultérieure, le contacter afin de lui demander la liste des équipements connectés qu'il connaît.

Publication: Il n'y a pas de publication des index car le réseau overlay est non structuré. Cependant, il est possible que certains nœuds puissent conserver temporairement des index dans un cache pour améliorer les performances de recherche.

Recherche: Lorsque les nœuds désirent effectuer une recherche de données, le routage de la requête se fait par inondation aux autres nœuds. Il existe plusieurs algorithmes d'inondations (par exemple, Breadth First Search, Depth First Search, et

contrôle par durée de vie (TTL)). Pour éviter d'augmenter le coût de recherche en cryptant/signant tous les messages, nous proposons d'émettre ces requêtes sans cryptage/signature. Une fois que le nœud destination est trouvé (par exemple, Alice), nous pouvons établir une session sécurisée de récupération des données en utilisant IBC [BoFr01] comme le montre la Figure 4.2.1-1.

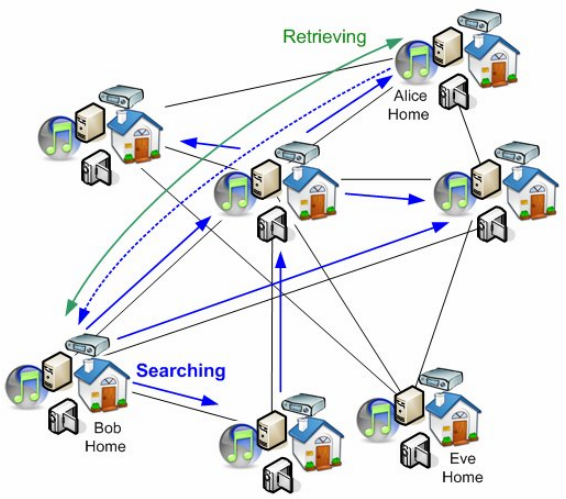

Figure 4.2.1-1: Exemple de recherche en P2P pur

Récupération: Connaissant l'emplacement exact des données, la récupération a lieu directement au nœud destination.

4.2.2. Proposition d'authentification avec IBC

Les architectures P2P pur pour le partage des ressources ne prévoient pas de mécanismes d'authentification forts. Cependant, le besoin global de sécurité dans l'Internet a apporté des mécanismes d'authentification. « Web of Trust » utilise le PGP (Pretty Good Privacy) [Zimm95] présenté dans le chapitre 2, il élimine le recours au serveur d'authentification centralisé avec toutefois des problèmes de confiance à accorder aux nœuds de confiance car la vérification de certificat dépend des utilisateurs.

Dans un contexte de réseau à domicile, la situation peut être un peu différente puisque la passerelle domestique peut être configurée par l'opérateur et être considérée comme digne de confiance. C'est pourquoi nous proposons d'utiliser la cryptographie par identité (IBC) [Sham85] et éviter ainsi les problèmes liés à la validation. Au début d'une communication, l'émetteur prend l'identité de l'utilisateur (ID) comme clé publique et élimine alors le recours à un serveur de certificats.

4.2.2.1. Identification de la signalisation

La Figure 4.2.2-1 présente les échanges de signalisation du service sécurisé. Les échanges s'effectuent sur le scénario, en 3 phases : enregistrement, recherche et récupération (il n'y a pas de publication). Nous explicitons ci-après la signalisation de chaque phase.

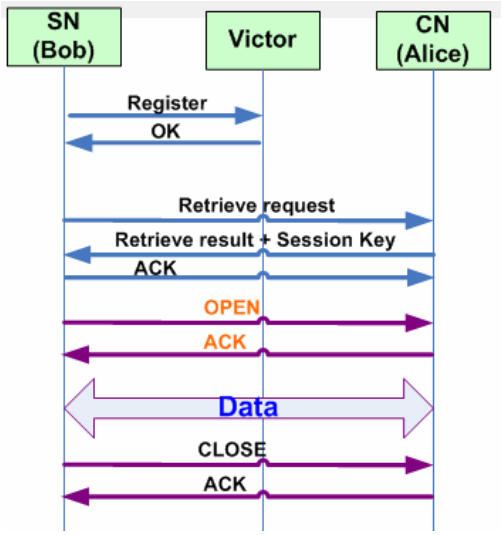

Figure 4.2.2-1: Principales étapes de signalisation dans une solution P2P pur

Enregistrement: Le nœud source (SN) se connecte à un de ses voisins (Victor connu par un technique d'amorçage préalable). Le message registre emis contient sa signature et est chiffré par l'identité de Victor. Le nœud source s'entregistre auprés de tous ses voisins (non representés dans la Figure 4.2.2-1). Deux messages de signalisation par voisin ont été échangés.

Recherche: Le nœud source transmet en inondation à ses voisins une requête à ses voisins. Ceux-ci transmettent à leur tour le message selon les spécificités de valeur Time-To-Live (TTL). Si la valeur maximale (prédéterminée par l'algorithme) est atteinte le message n'est pas retransmis. Le nombre de messages de signalisation est déterminé dans le paragraphe suivant.

Récupération: Une fois que SN a trouvé du contenu au nœud correspondant (CN), il prend sa clé privée pour signer et chiffrer les données avec l'identité du CN. SN envoie *Retrieve request*. CN renvoie *Retrieve result* avec une clé de session et sa signature. Ce message est chiffré par l'identité SN. SN vérifie le message et renvoie un message ACK au CN. SN envoie OPEN pour demander le partage des ressources qui peuvent également être chiffrées avec la clé de session précédemment obtenue. CN répond ACK à SN et la session est établie. Après cette session se termine par l'envoi des messages CLOSE et ACK. Le nombre de messages de signalisation est alors 7.

Exemple de protocole d'authentification

1) System parameters

$\{G_1, G_2, q, \hat{e}, P, P_{pub}, H_1, H_2\}$, les détails de ces paramètres référent à [BoFr01].

2) Registration

$\{SN, CN, \textit{regist}\text{ request, nonce, } SN_{sign}\}_{cn@example.com}$

$\{SN, CN, \textit{OK}\text{, nonce, } CN_{sign}\}_{sn@example.com}$

3) Searching

{SN, Broadcast, *search, keyword*}
4) Retrieving
{SN, CN, *retrieve request*, *URI*, nonce, SN_{sign}}$_{cn@example.com}$
{CN, SN, *retrieve result*, *session_key*, nonce, CN_{sign}}$_{sn@example.com}$
{SN, CN, *OK*, nonce, SN_{sign}}$_{cn@example.com}$
Secure tunnel: {CN, SN, *Open* || *Close* || *Ack* || *Content*}$_{session\text{-}key}$

« SN_{sign} » désigne la signature de SN, qui est crée en cryptant un message par sa clé privée. « {A}$_{b@example.com}$ » désigne le message « A », qui été crypté en utilisant la clé publique « *b@example.com* ».

4.2.2.2. Signalisation générée en phase de recherche

Des phases de signalisations, la phase de recherche est la plus lourde en raison de son fonctionnement par inondation. Nous en détaillons ci-après les échanges au travers du réseau.

L'algorithme d'inondation s'exécute étapes par étapes. A chaque étape nous transmettons une requête d'un nœud vers ses voisins. Le début de la transmission se fait au nœud racine. L'algorithme se termine lorsque le TTL est atteint. Puis, le nœud destination renvoie le résultat de la requête au nœud source (nœud racine) par le chemin de retour le plus court déterminé en utilisant l'algorithme de Dijkstra.

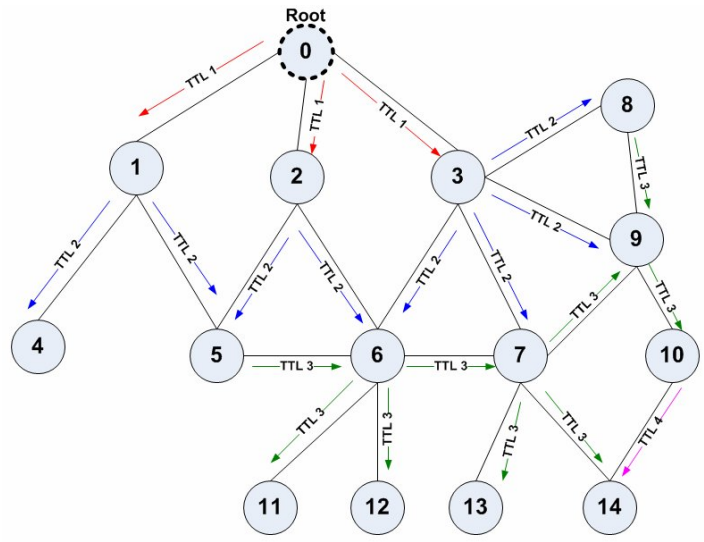

Figure 4.2.2-2: Exemple de recherche en inondations dans un réseau pair à pair

Par exemple, considérons le réseau de la Figure 4.2.2-2 et supposons que le nœud 1 est le nœud destination, celui qui possède la ressource, et que TTL = 4. L'algorithme d'inondation commence au nœud racine 0.

A l'étape 1: le nœud 0 envoie la signalisation à ses voisins : 1, 2 et 3.

A l'étape 2: le nœud destination 1 envoie un accusé de réception à la racine 0. Le nœud 0 calcule la distance qui le sépare de 1. Cette distance vaut 1.
A l'étape 3: les nœuds 1, 2 et 3 envoient les signalisations à leurs voisins qui à leurs tours envoient les signalisations à leurs voisins … jusqu'à ce que le TTL soit atteint.

Notez qu'un nœud récepteur ne transmet jamais de signalisation à un expéditeur qui lui a déjà envoyé celle-ci (par exemple, nœud 5 ne transmet pas au nœud 1 ou au nœud 2).

4.2.2.3. Synthèse des caractéristiques de déploiement

Nous synthétisons les caractéristiques de déploiement de l'architecture pair a pair pure ainsi que celles de la solution standard IMS et de la solution alternative SIP proxy dans la Table 4.2.2-1. L'IBC avec le P2P pur n'a pas besoin d'une clé pré-partagée, cependant, il doit avoir une clé privée qui correspond à son identité et aux paramètres du système. L'IBC peut facilement être intégré dans un environnement distribué. Quant à l'architecture SIP, elle utilise une clé pré-partagée pour réaliser la méthode HTTP Digest avec le serveur AAA; cette méthode est appliquée pour créer une clé de session qui sera utilisée entre l'utilisateur et son proxy. Le proxy peut être un pare-feu qui protégera le système contre les intrusions (par exemple, contre les attaques DoS) et permet d'en augmenter la sécurité. Du point de vue des calculs, la cryptographie symétrique utilisée dans SIP est moins complexe, que la cryptographie asymétrique utilisée dans l'IBC.

Il est à souligner que toutes les architectures de réseau proposées offrent des mécanismes d'authentification. Ceux-ci ne dépendent pas d'un algorithme de cryptage spécifique. L'administrateur de réseau choisit librement un protocole adapté (ex, RC4, 3DES).

Table 4.2.2-1: Hypothèse de sécurité pour le déploiement

	Pure P2P	**SIP Proxy**	**IMS**
Amorçage	Clé privée correspondent à l'ID et des paramètres.	Clé pré-partagée entre utilisateur et serveur AAA	Clé pré-partagée entre utilisateur et HSS
Eléments de confiance	PKG	proxy et AAA	CSCFs, AS et HSS
Mécanisme de sécurité	IBC Clé de session	HTTP Digest Clé de session	HTTP Digest IPSec

4.2.3. Modélisation et évaluation

Après avoir explicité le fonctionnement de l'architecture sécurisée en pair à pair pur nous analysons l'interêt de cette solution. Nous nous attachons en premier lieu à déterminer les paramètres de l'architecture, qui nous permettent de comparer une

architecture sécurisée en pair à pair distribuée à l'architecture en pair à pair centralisée que nous avons étudiée dans le chapitre précédent. Cet aspect de modélisation sera ensuite employé pour analyser dans une deuxième étape les mécanismes d'authentification que nous proposons pour sécuriser l'architecture. Pour mener à bien notre analyse, nous comparons la signalisation induite en rapport à celle générée par l'architecture de référence IMS ainsi qu'avec l'architecture SIP que nous avons proposée.

4.2.3.1. Caractéristiques topologiques du réseau P2P

Les performances d'un réseau pair à pair sont fortement influencées par le nombre d'éléments qui le composent et par la topologie du réseau d'overlay qui les relie.

Historiquement, de nombreuses questions se sont posées sur les moyens de décrire les topologies de systèmes complexes. Très tôt les réseaux complexes ont été décrits à l'aide des graphes aléatoires de Erdos et Renyi [ErRe85], où les nœuds sont les éléments du système et les liens représentent les connexions entre les nœuds. Cette modélisation est commune à de nombreux systèmes Par exemples (1) dans un réseau génétique, les protéines et les gènes sont des nœuds et des liaison chimiques sont des liens, (2) dans une organisation, les ministères sont des nœuds, et les communications entre les ministères sont des liens, et (3) dans les documents WWW, les documents HTML sont des nœuds et les hyperliens entre les documents sont des liens, etc. La représentation par graphe s'avère complexe sur les systèmes de grande taille où il est difficile d'obtenir l'information de raccordement, mais l'automatisation de l'acquisition des données peut permettre de résoudre ce problème. Des travaux ayant pour objet la collecte de données réelles sur les topologies de grands systèmes et leurs évolutions indiquent une modélisation possible par loi exponentielle. Ainsi les études empiriques [FaFF99][BaAl99] révèlent que la topologie d'Internet suit des lois exponentielles. Le réseau overlay de la communauté P2P pur suit également une loi exponentielle [ALPH01][JoAB01][AtGM03].

En ce qui nous concerne notre objectif est de disposer de topologies de réseaux P2P pour tester des scénarios de service, évaluer des solutions de sécurité. Nous devons connaître le nombre de nœuds, le nombre de degré de chaque nœud, et le lien entre les nœuds. Pour ce faire nous avons utilisé un outil de génération de topologie.

4.2.3.2. Le générateur de topologie: BRITE

Plusieurs générateurs de topologies (par exemple, Waxman [Waxm88], Inet-3.0 [JiCJ00], GT-ITM [CaDZ97] et BRITE [MLMB01]) ont été développés dans le cadre de travaux de recherche. Ils reposent soit sur des modèles ad hoc tel que Waxman qui modélise la connectivité entre paire de nœuds en se référant à la distance qui les sépare, soit sur des modèles établis à partir de mesures comme par exemple la modélisation par loi de puissance du degré de sortie d'un nœud. (Cette loi s'est avérée être proche de la représentation de l'Internet). Parmi ces derniers, les auteurs de [MLMB01] identifient les modèles qui appliquent une loi de puissance et ceux qui appliquent une heuristique pour faire croître le réseau telle « l'ajout d'un nœud dans la topologie se fait en le raccordant à un nœud ayant le plus haut degré» (ex : modèles de Barabasi-Albert model [BaAl99]).

En ce qui nous concerne nous utilisons pour ce travail l'outil BRITE, c'est un outil disponible sur Internet qui offre plusieurs moyens de représenter des systèmes. BRITE a été élaboré en 2001 pour générer des topologies reposant sur la loi exponentielle. Il introduit également des observations de placement des nœuds et de localisation dans les connexions réseau sur l'Internet. BRITE a été utilisé dans le contexte de routage [KuPo08], BGP [CMYP07], et est intégré à d'autres outils d'études de topologie dans le travail présenté dans [RaPa05].

Principes de BRITE
BRITE est un générateur de topologie Internet qui prend en charge les modèles de plusieurs niveaux de topologie : niveau système autonome (AS: Autonomous System), niveau routeur. En fait, le niveau AS est assez similaire au niveau du routeur, la principale différence est que les nœuds AS ont la capacité de contenir des topologies associées.

Caractéristiques de BRITE
BRITE est conçu pour être un générateur de topologie flexible qui produit des données pour des simulateurs de réseaux (ex, SSF, JSIM et NS-2). Il utilise une architecture orientée objet qui fournit aux utilisateurs la possibilité d'ajouter de nouveaux modèles de production et la possibilité d'importer et exporter des fichiers personnalisés de topologies. Il permet également l'importation d'autres générateurs de topologies en les étendant ou en les combinant. Il fournit à l'utilisateur une interface graphique (GUI) et un fichier de configuration pour spécifier facilement divers paramètres du générateur de topologie. En outre, il y a d'autres paramètres pour générer des topologies qui permettent d'améliorer les modèles, par exemple, le placement de nœuds, la bande passante, les délais.

Actuellement, BRITE ne dispose que d'une seule interface de résultat. Il crée un fichier de topologie avec un format du fichier choisi par l'utilisateur. Cet outil n'a pas d'outil de visualisation associé.

4.2.3.3. Modélisation de la topologie et des coûts de signalisation

Pour évaluer la signalisation en architecture pair à pair pur, contrairement au pair à pair centralisé, il est nécessaire de connaître le nombre d'utilisateurs et leur localisation, c'est-à-dire la topologie du réseau physique et overlay.

1) Modélisation de topologie et méthode d'évaluation

Les caractéristiques du modèle P2P sont le nombre de nœuds et la distribution des interconnexions. La distribution des degrés de nœud peut-être modélisée en utilisant la loi de puissance [JoAB01]. Des exemples de réseaux réels suivant cette distribution sont identifiés dans [Jova01].

Lorsqu'un nœud *i* rejoint le réseau, la probabilité qu'il se connecte à un nœud *j* qui est déjà connecté aux autres nœuds est donnée en (4.2.3-1), où *d* est le degré du nœud cible, *V* est l'ensemble des nœuds qui sont déjà membres du réseau et $\sum_{k \in V} d_k$ est la somme de leurs degrés

$$P(i,j) = \frac{d_j}{\sum_{k \in V} d_k} \qquad (4.2.3\text{-}1)$$

Nous supposons que la taille du réseau physique est 1000 nœuds connectés (passerelles résidentielles et routeurs) et que l'effectif des participants est de 100 utilisateurs connectés. Nous avons généré une topologie P2P pur en utilisant l'outil BRITE [MLMB01] avec le modèle de Barabasi-Albert [BaAl99]. Il crée un graphe qui comporte 1000 nœuds pour le réseau physique et *n* nœuds pour un réseau overlay suivant la loi de puissance. À partir de ce réseau physique, 100 topologies overlay de degré minimum = 1 et 100 autres de degré minimum = 2 sont générées pour chaque valeur de *n*. Ces nœuds sont choisis au hasard parmi les nœuds de degré 1 du réseau physique.

Nous avons évalué la signalisation générée par l'algorithme de recherche en inondation et également évalué la distance exprimée en nombre moyen de hop. Pour l'inondation, le TTL a été fixé à 4 ce qui correspond à la configuration par défaut fixée pour un client du réseau P2P Gnutella.

Pour calculer le coût moyen de la signalisation de recherche nous avons testé toutes les paires émission destination (avec variation des nœuds sources et des nœuds destination jusqu'à atteindre tous les nœuds).

En conséquence nous obtenons des valeurs pour S_n^{cost}, le coût de signalisation pour un réseau P2P de *n* clients correspondant au coût de signalisation de la recherche pour *n* nœuds reliés dans le réseau overlay (*n* variant de 10 à 100 par incréments de 10 nœuds pour chaque étape).

En ce qui concerne la distance en nombre de nœuds traversés, pour évaluer le coût dans la phase de récupération, dans un premier temps les expéditeurs et les récepteurs sont choisis aléatoirement dans le réseau overlay de simulation. Puis, le routage basé sur l'algorithme de Dijkstra est appliqué résultant en une distance moyenne de nombre de nœuds entre le SN et le CN. *Cette valeur moyenne est trouvée égale à 7.*

2) Formulation des coûts de signalisation

Nous formulons le coût de signalisation de l'architecture proposée ainsi que ceux des architectures IMS et SIP proxy de façon à obtenir un même cadre comparatif. Le coût dépend du nombre d'échanges de signalisation et de la distance entre la source et la destination de l'échange de service qui a été explicité dans la section précédente (valeur 7). Il s'agit également de considérer les paramètres ci-après :

Soit un taux d'utilisation tel que, les utilisateurs actifs (N) représentent 50% des *n* nœuds connectés. Ainsi, *N* varie de 5 à 50 dans les simulations.

Notons *R* le taux moyen d'utilisation de transaction pour chaque phase. Nous fixons la valeur de R à 1 et à 5 dans les simulations.

Soit γ le coût associé à la transmission d'un paquet sur un lien. Et $d_{SN\text{-}CN}$ désigne la distance moyenne entre le nœud source (SN) et le nœud correspondant (CN).

Nous formulons alors la signalisation pour les trois architectures comme suit :

2.1) Architecture P2P pur avec IBC

Le coût de l'ensemble de la signalisation en utilisant P2P pur est donné par:

$$C_{total}^{IBC} = C_{register}^{IBC} + C_{publish}^{IBC} + C_{search}^{IBC} + C_{retrieve}^{IBC}$$

Où, chaque phase de transmission de signalisation est donnée par:

$$C_{register}^{IBC} = N^{*}R_{reg}\,(\gamma\,(\,2\ d_{SN\text{-}CN}\,))$$

$$C_{publish}^{IBC} = 0 \quad \text{(les nœuds gardent les index, il n'y a pas de publication)}$$

$$C_{search}^{IBC} = N^{*}R_{sea}\,(\gamma\,(S_n^{cost}\ d_{SN\text{-}CN}\,))$$

$$C_{retrieve}^{IBC} = N^{*}R_{ret}\,(\gamma\,(\,7\ d_{SN\text{-}CN})\,)$$

Nous pouvons remarquer que le pourcentage d'éléments actifs dans le réseau d'overlay (N) n'influe pas sur la forme des résultats. Ceux obtenus avec le pourcentage de 50% que nous avons fixé seront similaires à ceux qui seraient obtenus avec une valeur de pourcentage différente.

2.2)Architecture P2P SIP proxy

Rappelons le fonctionnement de l'architecture proxy en se référant à la Figure du chapitre 3 (Figure 4.2.3-1) ci-après pour la phase d'enregistrement.

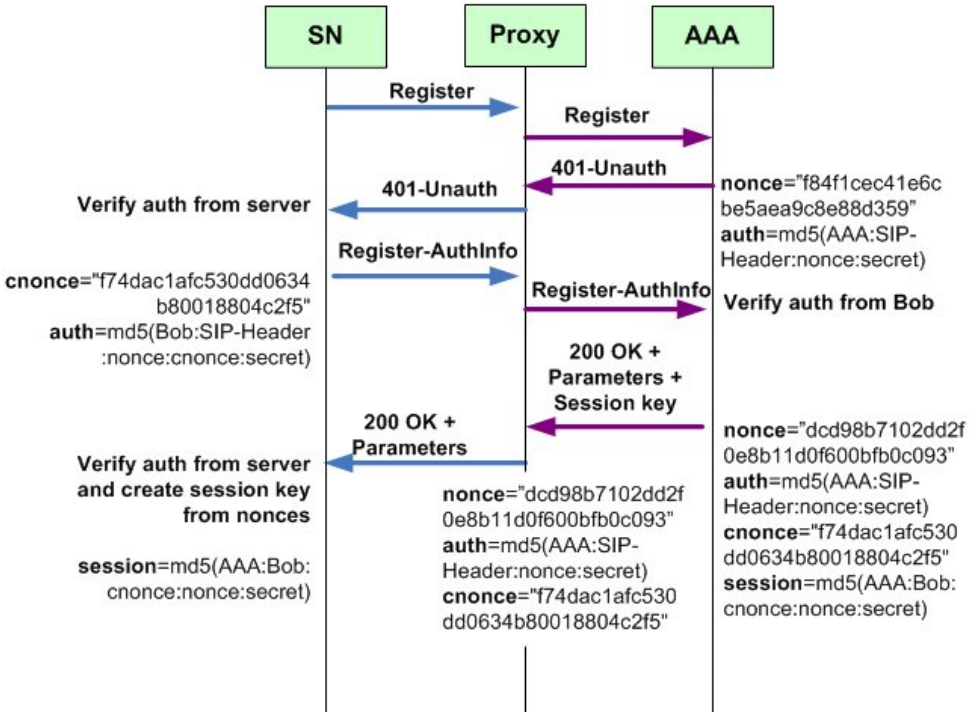

Figure 4.2.3-1: Signalisation d'enregistrement de l'architecture SIP

Ainsi qu'à la figure suivante (Figure 4.2.3-2) pour les phases de publication recherche et récupération.

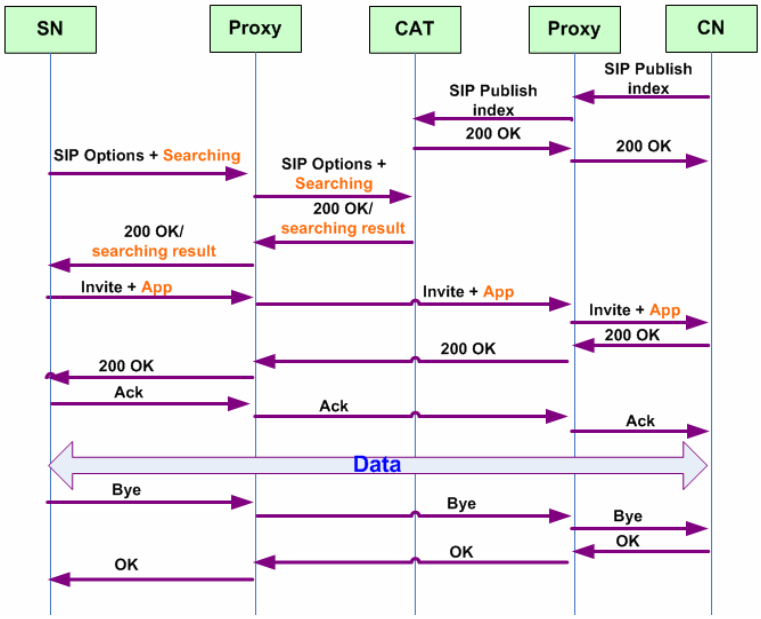

Figure 4.2.3-2: Signalisation de l'architecture SIP proxy en phase de publication, recherche et récupération

Le coût de l'ensemble de la signalisation en utilisant la solution SIP est alors donné par:

$$C_{total}^{SIP} = C_{register}^{SIP} + C_{publish}^{SIP} + C_{search}^{SIP} + C_{retrieve}^{SIP}$$

Où, le coût de chaque phase de transmission de signalisation est exprimé par:

$$C_{register}^{SIP} = N{*}R_{reg}\,(\gamma\,(\,4d_{\text{SN-Proxy}} + 4d_{\text{Proxy-AAA}})\,)$$

$$C_{publish}^{SIP} = N{*}R_{pub}\,(\gamma\,(\,2d_{\text{CN-Proxy}} + 2d_{\text{Proxy-Cat}}\,)\,)$$

$$C_{search}^{SIP} = N{*}R_{sea}\,(\gamma\,(\,2d_{\text{SN-Proxy}} + 2d_{\text{Proxy-Cat}})\,)$$

$$C_{retrieve}^{SIP} = N{*}R_{ret}(\gamma\,(\,5d_{\text{SN-Proxy}} + 5d_{\text{Proxy-Proxy}} + 5d_{\text{Proxy-CN}})\,)$$

avec $d_{x\text{-}y}$ désigne la distance en nombre de sauts de x à y, où x et y sont des éléments de l'architecture SIP. Comme par exemple le proxy et le serveur d'authentification.

2.3) L'IMS avec AKA

Nous formulons de façon similaire le coût de l'ensemble de la signalisation en utilisant IMS par:

$$C_{total}^{IMS} = C_{register}^{IMS} + C_{publish}^{IMS} + C_{search}^{IMS} + C_{retrieve}^{IMS}$$

Où, chaque phase de transmission de signalisation correspond à :

$$C_{register}^{IMS} = N{*}R_{reg}\,(\gamma\,(\,8d_{\text{SN-Pcscf}} + 4d_{\text{Pcscf-Icscf}} + 4d_{\text{Icscf-HSS}} + 4d_{\text{Icscf-Scscf}} + 4d_{\text{Scscf-HSS}})\,)$$

$$C_{publish}^{IMS} = N{*}R_{pub}\,(\gamma\,(\,7d_{\text{CN-Pcscf}} + 7d_{\text{Pcscf-Scscf}} + 7d_{\text{Scscf-AS}}\,)\,)$$

$$C_{search}^{IMS} = N{*}R_{sea}\,(\gamma\,(\,7d_{\text{SN-Pcscf}} + 7d_{\text{Pcscf-Scscf}} + 7d_{\text{Scscf-AS}}\,)\,)$$

$$C_{retrieve}^{IMS} = N{*}R_{ret}\,(\gamma\,(7d_{\text{SN-Pcscf1}} + 7d_{\text{Pcscf1-Scscf1}} + 2d_{\text{Scscf1-Icscf}} + 5d_{\text{Scscf1-Scscf2}} + 2d_{\text{Icscf-HSS}} + 2d_{\text{Icscf-Scscf2}} + 7d_{\text{Scscf2-Pcscf2}} + 7d_{\text{Pcscf2-CN}}\,)\,)$$

Le d_{x-y} désigne distances de hop de x à y, qui sont des éléments de l'architecture IMS.

4.2.3.4. Simulation et évaluation

Les valeurs des paramètres de l'évaluation sont indiqués ci-après.

La taille des paquets est simulée par une distribution *trunked Pareto* avec une moyenne égale à 480 octets [TaFB06]. Le débit de données est considéré pour une capacité de 10 Mbps, soit 3,84 x 10-4 sec.

D'après les résultats de simulation sur Brite, la distance en nombre de nœuds entre SN et CN est en moyenne de 7 nœuds. Dans SIP et IMS, la distance entre l'utilisateur et le réseau de service (e.g., $d_{S/CN-Proxy}$ sur le SIP, $d_{S/CN-P-CSCF}$ sur l'IMS) est de 3, avec 2 routeurs en moyenne sur le chemin, tandis que la distance à l'intérieur du réseau de service (e.g., $d_{Proxy-Proxy}$ sur le SIP, et d_{x-CSCF} d_{y-CSCF} sur l'IMS) est fixé à 2 [MuVW08].

Bien que les plages de valeurs des paramètres soient représentatives de situations réelles, d'autres modélisations, en particulier du P2P, produiraient des valeurs différentes. Les tendances des résultats que nous présentons seraient pourtant similaires.

Considérons dans un premier temps l'influence de la taille du réseau physique dans la Figure 4.2.3-3. Sans analyser exactement les résultats nous pouvons voir une tendance générale similaire : le coût de la solution P2P pure est toujours plus important que celui des 2 autres solutions, et celui de IMS est plus important que le SIP proxy. Au vu du contexte des travaux il paraît plus adapté de faire des simulations sur un grand nombre de nœuds. Par la suite nous présentons les résultats pour 1000 nœuds.

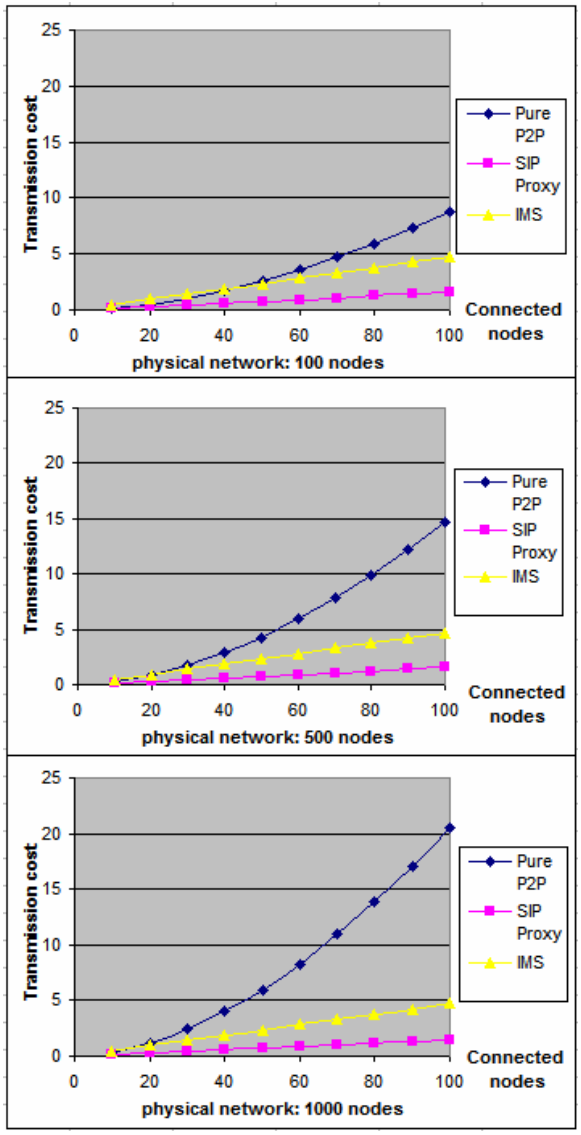

Figure 4.2.3-3: Importance de la taille du réseau physique sur la signalisation

La Figure 4.2.3-4 et la Figure 4.2.3-5 indiquent le coût de communication qui correspond aux nœuds actifs et celui de l'établissement de la connexion en fonction du nombre de nœuds connectés. Il y a l'enregistrement, la publication, et UNE recherche puis récupération par nœud actif (R_{sea} = 1 pour toutes les phases) dans la Figure 4.2.3-4, et nous considérons CINQ recherches (R_{sea} = 5) dans la Figure 4.2.3-5.

Conformément à la formule, le coût de transmission augmente avec le nombre d'utilisateurs actifs. Lorsque le processus est basé sur SIP, le coût de transmission est minimal. Le P2P pur présente le coût le plus élevé, en raison de l'algorithme d'inondation utilisé dans la phase recherche (un algorithme plus sophistiqué pourrait être adopté). En outre, la distance moyenne est plus grande que dans les autres cas. L'écart entre les coûts de transmission d'IMS et de SIP diminue lorsque le nombre de recherches par utilisateur augmente. Ainsi, pour un service fréquemment utilisé, IMS et SIP n'ont pas des performances très différentes en comparant au P2P pur.

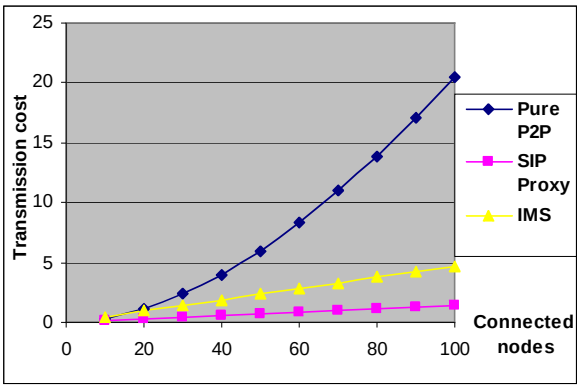

Figure 4.2.3-4: Coût de transmission pour 1 enregistrement, publication, et 1 recherche récupéreration, par utilisateur

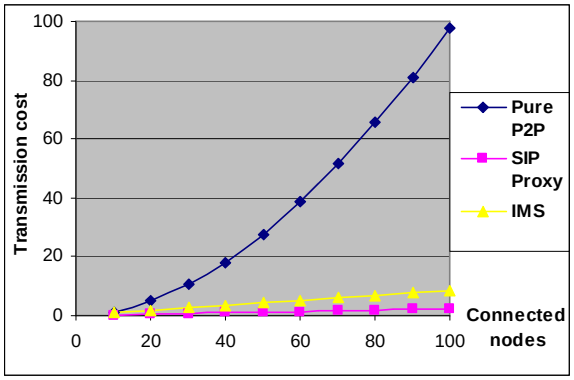

Figure 4.2.3-5: Coût de transmission pour 1 enregistrement, publication, et 5 recherches/récupérations par utilisateur.

Les composantes de la signalisation sont détaillées sur la Figure 4.2.3-6. Comme présenté dans le chapitre précédent, la signalisation SIP proxy est plus faible que celle d'IMS, car l'architecture ne traite pas tous les aspects de la norme IMS, (les informations sont incluses dans les messages SIP au cours de la publication et la recherche), il n'a pas en SIP proxy de message SIP INVITE en phase de Publication et Recherche et il y a moins de composants que dans l'architecture IMS. Le coût SIP est également nettement inférieur au coût d'une signalisation pair à pair pur.

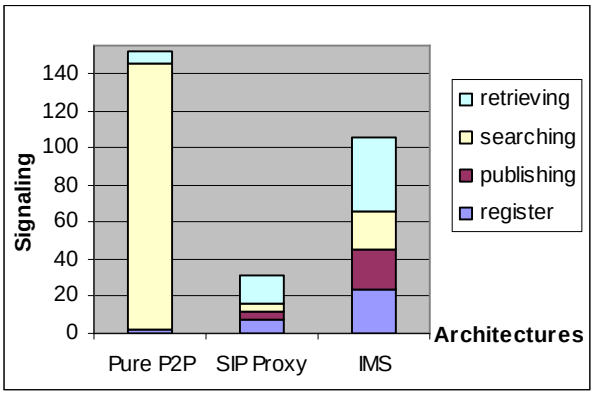

Figure 4.2.3-6: Signalisation dans les architectures de réseau pour une recherche

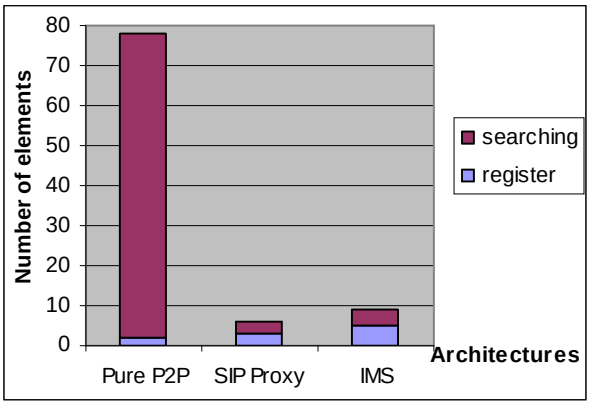

Figure 4.2.3-7: Nombre d'éléments de l'architecture participant aux principales étapes du service

Le coût majeur dans le P2P est dû à la recherche. Néanmoins ce coût doit être relativisé car bien que le P2P pur ait plus de signalisation par rapport à SIP, les nœuds individuels n'ont pas à gérer un trop grand trafic de signalisation. Pour se rendre compte de la part de traitement par nœud nous exprimons le nombre d'élements impliqués dans chaque architecture dans la Figure 4.2.3-7. Nous voyons qu'il y a beaucoup de nœuds qui travaillent ensemble de façon décentralisée pour le partage des ressources. La quantité de signalisation par nœud peut ne pas être alors très différente de celle induite par la solution SIP proxy. D'autre part, le serveur de catalogue centralisé dans SIP et IMS gère de nombreuses requêtes, contrairement au P2P pur.

Ainsi cette architecture n'est pas si coûteuse pour un utilisateur individuel, lorsque le nombre de participants augmente.

4.2.4. Résultats

En analysant le coût de la solution en terme de signalisation, nous constatons que l'architecture centralisée à base de SIP génère moins de signalisation tout en proposant un bon niveau de sécurité. En outre, le proxy SIP qui est en charge du routage des messages SIP pourrait être également utilisé comme un pare-feu pour se protéger contre des intrus.

Une autre remarque concernant le coût de la solution est liée à la fréquence d'utilisation du service (le nombre de *Recherche*s effectuées par connexion au service). Alors que les performances de solution classique à base d'IMS s'avéraient être correcte devant celle en SIP proxy, pour des utilisations fréquentes de services, en ce qui concerne la solution P2P pur, la signalisation est trop lourde.

L'architecture pair à pair pure ne semble être déployable que dans de petits réseaux, limités à un petit cercle d'amis. Cependant nous devons prendre en compte les possibilités d'optimisation de la phase de recherche. Dans la section suivante nous

allons considérer une solution pair à pair moins onéreuse sur la phase de recherche qui considère un mécanisme par table de hachage.

4.3. Architecture P2P avec DHT

4.3.1. RELOAD pour les services à domicile

Un moyen d'améliorer les performances de la phase de recherche est l'utilisation d'un algorithme reposant sur des tables de hachage et non de la diffusion globale. La solution que nous étudions est une architecture P2P distribuée avec table de hachage (Distributed Hash Table) qui s'appuie sur un pré-calcul des positions de nœuds ou des ressources dans un réseau overlay. Pour cette architecture nous nous appuyons sur des les travaux en cours au sein de l'IETF : *Resource Location And Discovery* (RELOAD) [JLRB12] est un protocole de signalisation pour P2P qui inclut des mécanismes de sécurité et repose sur les algorithmes de Chord.

Nous allons expliciter l'architecture P2P DHT « RELOAD » pour les services à domicile, puis nous décrirons la preuve de concept que nous avons effectuée.

4.3.1.1. Caractéristiques de RELOAD

RELOAD fournit un service d'échange de message et de mémorisation abstraite à des nœuds coopérants dans un réseau d'overlay, en s'appuyant sur la signalisation SIP et l'algorithme Chord pour construire son réseau overlay. La notion principale de l'architecture est l'identification. Chaque nœud du réseau overlay est identifié par un identificateur de 128 bits, qui est appelé *Node-ID*. Les ressources sont enregistrées comme des adresses numériques qui occupent le même espace que l'identifiant *Node-ID*. Le *Node-ID* est utilisé principalement pour trois raisons :

- pour référencer le nœud lui-même.
- pour déterminer sa position dans la topologie overlay quand elle est structurée.
- pour déterminer l'ensemble des ressources dont le nœud est responsable.

Node-ID peut être tout simplement généré par hachage (Hash(X) : où X peut être l'adresse IP, la clé publique), cependant il est généralement attribué à partir d'un serveur d'inscription central afin d'éviter les attaques de type Sybil [Douc02]. Chaque nœud possède un certificat [CSFH08] contenant le *Node-ID* qui est unique dans une instance de réseau overlay (Nous abordons la génération de certificat dans le paragraphe authentification).

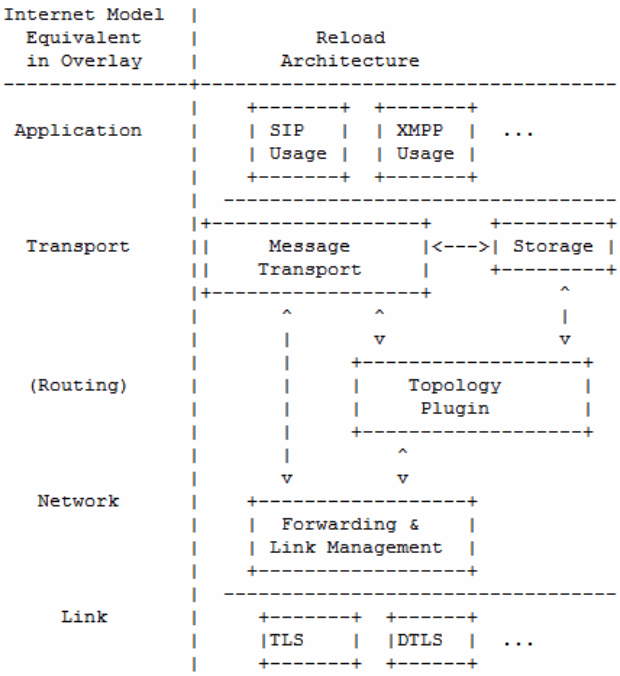

Figure 4.3.1-1: L'architecture RELOAD IETF

Les composants principaux (Figure 4.3.1-1) de RELOAD tel que présenté dans l'internet draft [JLRB12], sont les suivants:

Usage Layer: il définit la façon d'utiliser le service de RELOAD. Il supporte actuellement comme protocoles applicatifs SIP, et XMPP

Message Transport: il est en charge de la fiabilité de bout en bout, du traitement des états des requêtes ainsi que du relayage des opérations de stockage et extraction sur les composants de mémorisation. Ce composant délivre une réponse à un élément qui a initié une requête.

Storage: pour le traitement des messages relatifs au stockage et à la récupération de données. La gestion de la réplication et de la migration des données se fait par un dialogue avec la composante *Plugin topology*.

Topology Plugin: cet élément implante l'algorithme de réseau overlay. Il utilise le composant Message Transport pour émettre et recevoir des messages de gestion du réseau overlay, les messages sont passés aux composants *Storage* lorsqu'ils concernent la replication des données et dirigés vers le *Forwarding and Link Mangement Layer* pour contrôler le relayage des messages.

Forwarding and Link Management Layer: il gère la table de routage et l'établissement de nouveaux liens entre les nœuds (y compris l'aspect NAT). La table de routage (*Finger Table)* contient les informations pour router les messages de recherche (*Finger)*

Overlay Link Layer: pour le transport du trafic entre les nœuds. Permet de fournir des mécanismes d'acquittement en cas de transport non fiable. TLS [DiRe08] et DTLS [ReMo06] sont les protocoles de sécurité actuellement supportés.

RELOAD est encore en phase de recherche. Parmi les travaux dont il fait l'objet, citons [BrLZ08] qui l'utilise pour une communication sécurisée à l'Internet public, en

utilisant le certificat, et le routage dans un environnement NAT. [YuCS10] essaie d'augmenter la disponibilité du service dans le P2P pour les nœuds qui ont des ressources limitées en proposant un nouveau algorithme de construction du réseau overlay. La performance de RELOAD sur téléphone mobile est analysée dans [MaBo10], et [KoBu11] propose des extensions de la couche application multicast.

4.3.1.2. L'authentification dans RELOAD

L'authentification de RELOAD part de l'hypothèse que chaque nœud possède un ou plusieurs certificats de clé publique. L'utilisateur doit obtenir la clé publique/privée et un certificat avant de rejoindre le réseau overlay. Le certificat vérifie le nom d'utilisateur et les *Node-IDs* dans le réseau overlay.

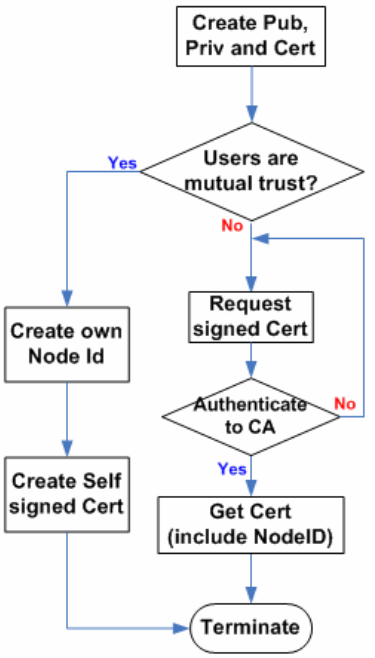

Figure 4.3.1-2: Génération de clé publique/privée et de certificat

La Figure 4.3.1-2 illustre l'algorithme de génération de la clé publique/privée et du certificat signé. L'utilisateur génère sa paire de clés publique/privée et le certificat. S'il communique avec des partenaires dans lesquels il a confiance, il peut créer ses *Node-IDs* (par exemple, en utilisant le hachage de sa clé publique afin de prévenir le vol de *Node-Id*) et signer lui-même son certificat. Sinon, un certificat signé doit être demandé à un serveur d'inscription central (connu par la configuration par défaut). Ce dernier attribuera ensuite un nom unique et un *Node-ID* qu'il inclura dans le certificat. Tous les nœuds d'un certain réseau overlay considèrent le serveur central d'inscription comme un point de confiance.

Une fois obtenu le certificat, trois niveaux d'authentification sont proposés aux nœuds pour vérifier l'origine et l'exactitude des données qu'ils reçoivent :

- *Niveau de la connexion*
  - Une fois qu'un nœud se connecte à un overlay, la négociation TLS/SSL a lieu avant d'envoyer des messages de service
- *Niveau du message*
  - Chaque message doit être signé à partir de la clé privée de l'expéditeur afin de confirmer son origine.
- *Niveau de l'objet*
  - Les objets stockés doivent être signés à partir de la clé privée du nœud les ayant publiés afin de vérifier le propriétaire de la ressource.

Notons que dans une architecture centralisée dans la mesure où les clients ont confiance dans le serveur de catalogue il n'y a pas de sécurisation au niveau objet. Dans le chapitre précédent, nous avons proposé de mettre en place uniquement une sécurité de niveau connexion.

Le fonctionnement général de cette architecture est alors :

Enregistrement : Pour joindre le réseau pair à pair, il faut envoyer son identité au nœud d'amorçage et obtenir sa position. Ensuite, il y a une vérification mutuelle pour établir une session sécurisée. Après cela, le nœud qui s'enregistre cherche ses voisins proches (par exemple, successeur, prédécesseur) qui seront utilisés pour le routage.

Publication: Un nœud qui souhaite partager une ressource crée un index par hachage des mots-clés (ex, le titre, le nom de fichier) pour en générer l'identité dans l'anneau Chord. (par exemple, l'index: {*4, p2psip://147.127.240.90/Alice.jpg*}, « *4* » été généré par Hash(Alice.jpg)). Puis, ces identités sont routées vers un nœud qui est en charge de cet index et le stocke. Notons qu'un mot-clé et une ressource sont en relation multiple (*many-to-many)*, ainsi un mot-clé peut caractériser de nombreuses ressources et une ressource peut être liée à plusieurs mots-clés.

Recherche: Un nœud fait le hachage des mots-clés pour générer l'identité qui identifiera le nœud Chord en charge du stockage de l'index de la ressource. A partir de ce nœud, il télécharge une liste d'adresses de ressources.

Récupération: Un nœud établit directement une session vers le nœud propriétaire de la ressource afin de la télécharger.

Une illustration du fonctionnement de l'architecture est fournie en Figure 4.3.1-3. Les nœuds N1, N3, N5, N7 et N12 rejoignent le réseau overlay. Le nœud N12 (IP:147.127.240.90) publie la ressource (*Alice.jpg*). Il crée un index et le signe avec sa clé privée. Il recherche ensuite dans sa table de routage overlay (*Finger table)* un nœud (ex, N3) en charge de cet index. Une fois la destination identifiée, un message RELOAD est créé et également signé avec sa clé privée. L'index est envoyé en utilisant un tunnel TLS/SSL à N1 puis transmis à N3. Ensuite, l'index est stocké en N3. N3 peut vérifier l'index de N12 avec la clé publique de N12.

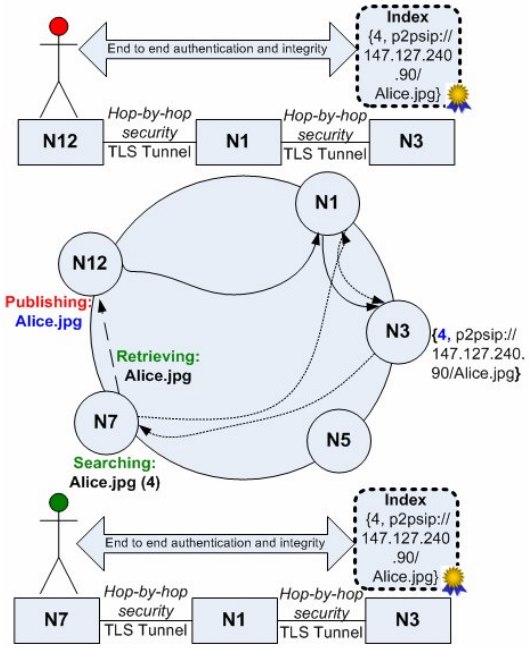

Figure 4.3.1-3: Exemple de délivrance de services sécurisé avec RELOAD

Le nœud désirant télécharger une ressource (N7) recherche *Alice.jpg* en créant l'identité de son mot-clé. Comme dans l'étape de publication, il cherche dans sa *Finger table* le nœud en charge de son mot-clé. L'identité est transmise jusqu'à N3, de qui N7 télécharge un index. Ensuite, N7 vérifie l'index en utilisant la clé publique de N12. Si l'index est correct, il peut télécharger directement de N12 qu'il connaît par l'index.

4.3.2. Expérimentation

Afin de prouver le concept de notre architecture, nous avons déployé RELOAD pour offrir le service de partage de photos. Dans cette section, nous décrivons les caractéristiques de notre système ainsi que des éléments relatifs à l'implantation que nous avons réalisée en Java, avant d'en présenter l'évaluation dans la section suivante.

4.3.2.1. Conception

Notre conception en couche de l'architecture RELOAD est illustrée à la Figure 4.3.2-1. Les principaux composants sont regroupés en trois couches nommées: *Application*, *RELOAD*, et *Transport*.

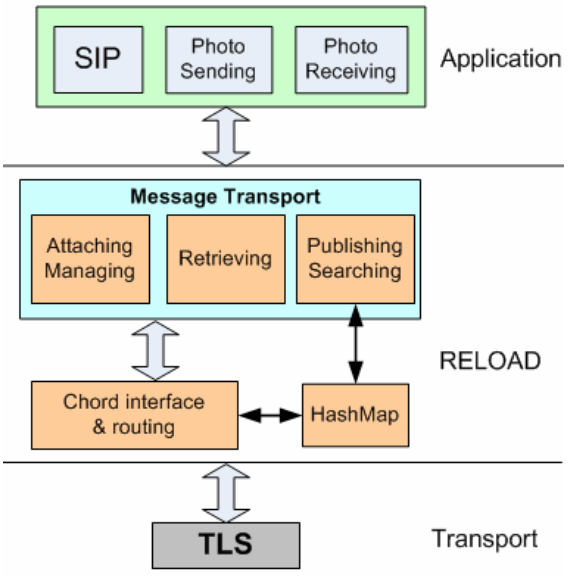

Figure 4.3.2-1: Principaux composants déployés pour notre architecture

1) Application layer

Comme défini dans le document IETF de RELOAD, nous utilisons un composant SIP pour la couche d'application. Deux autres composants, « envoi de photos » et « récupérer des images », sont également créés pour une réalisation simple du service de photo.

2) Reload layer

Cette couche est composée de trois éléments principaux :

a) Le composant *Message transport* : il gère le point d'entrée, utilise les méthodes correspondantes dans l'interface Chord, et le routage afin de satisfaire les étapes de service. Il se compose des composants : *Attaching/Managing*, *Retrieving* et *Publishing/ Searching.*

b) Le composant *Chord interface and routing* joue le rôle des composants *Topology plugin* et *Forward and link management* du document IET, qui gèrent les méthodes du routage Chord.

c) Le composant *HashMap* implémente le composant de stockage. Il stocke les objets d'un nœud *Chord*, mémorisés par des index de ressources (par exemple, l'URI) et une signature signée (par un propriétaire de la ressource). Ce composant est utilisé dans les phases de *Publishing* et *Searching*.

3) Transport layer

Nous avons choisi d'implanter TLS sur TCP par opposition à la version sur UDP (D-TLS) pour des contraintes de réalisation (disponibilité des bibliothèques Java). Nous n'avons pas choisi le mécanisme de IPsec que nous avons cependant utilisé pour la solution proxy (entre les serveurs). En effet dans la solution SIP proxy, les serveurs sont connus ce qui n'est pas le cas des nœuds Chord, ce qui rend plus complexe la phase d'amorçage.

Les principales fonctions que nous avons conçues sont :

Joining: contient les fonctions *FindSuccessor, ReturnSuccessor* utilisées pour trouver le successeur, *FindPredecessor, ReturnPredecessor* pour trouver le prédécesseur, *Stabilizing* et *NotifyPredecessor* pour mettre à jour son successeur et le prédécesseur de son successeur.

Stabilizing: inclus les fonctions *FindPredecessor, ReturnPredecessor and NotifyPre-decessor*. Elles sont utilisées en phase d'insertion (Joining, pour les deux premières) mais également de façon périodique pour gérer l'arrivée et la sortie de nouveaux nœuds.

Fix Finger Table: contient, *FindPredecessor, ReturnPredecessor, FindSuccessor-UpdateFinger* et *ReturnSuccessorUpdateFinger*, fonctions effectuées de façon périodique pour la mise à jour du routage des nœuds Chord à l'intérieur de la *Finger table* dans chaque objet de *Finger*.

Publishing: *LookUpNode, ReturnLookUpNode*. Ces fonctions sont utilisées pour trouver un nœud qui va conserver des index et *WriteResource* pour mettre l'index à jour sur un nœud destination.

Searching: *LookUpNode, ReturnLookUpNode,* pour localiser un nœud qui contient des index correspondants à des ressources; *DownloadResourceURI* et *ReturnResource-URI* pour télécharger l'index d'un nœud destination.

Retrieving: utilise le protocole SIP pour établir une session de données, ouvrir une port pour les données entrantes, et terminer la session (INVITE, 200 OK, ACK, BYE and ACK).

4.3.2.2. Réalisation

Le langage Java a été utilisé pour la mise en œuvre de l'architecture en raison de la disponibilité de ses bibliothèques. Sont disponibles, les éléments TLS/SSL, SIP et plusieurs bibliothèques de stockage. Cependant, il y a quelques inconvénients. En effet, RELOAD utilise des structures de données du langage C qui ne sont pas faciles à manipuler en Java, néanmoins la bibliothèque de javolution.org permet de résoudre ce problème. En outre, RELOAD est un protocole orienté message avec des messages encodés au niveau bit, peu pratique à mettre en œuvre en Java (il n'y a pas de type d'entête du style « nom-valeur » comme celle de SIP ce qui fait que le message doit être entièrement examiné avant de faire quoi que ce soit).

Simplification

RELOAD nécessite l'installation de l'Interactive Connectivity Establishment (ICE) [Rose10] ou ICE-Lite. Cette exigence est compréhensible pour une utilisation sur un réseau ouvert comme l'Internet dont certains nœuds peuvent avoir une adresse IP privée. Cependant, nous avons estimé qu'en plus du coût de sa mise en ouvre, ICE n'était pas nécessaire pour notre preuve de concept. En outre, nous mettons en œuvre uniquement les niveaux connexion et sécurité des objets. La sécurité de niveau connexion est similaire à celle mise en œuvre dans une architecture centralisée et celle de niveau objet remédie au problème de non confiance dans les nœuds sur lesquels est distribué le catalogue des ressources.

Implémentation

Application layer : nous mettons en œuvre les composants *SIP*, *Photo Sending* et *Retrieving*. SIP est utilisé pour ouvrir le port de données et accepter les flux de données entrant dans l'étape de récupération.

RELOAD layer, les nœuds communiquent en échangeant des messages définis en XML qui sont liés à des classes de « *Message Chord* » du type Plain Old Java Object (POJO). Un exemple de POJO est montré dans la Figure 4.3.2-2.

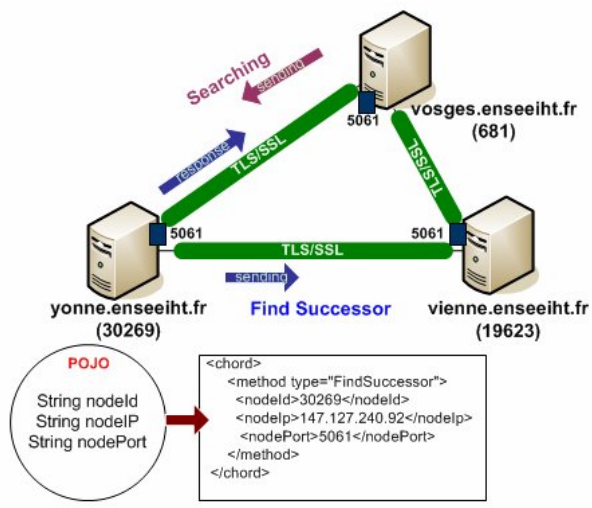

Figure 4.3.2-2: Exemple d'échanges sur le banc de test

La Figure 4.3.2-3 illustre la structure de la classe principale (ChordNode). Ce diagramme est lié à notre conception de la couche RELOAD comme le montre la Figure 4.3.2-1.

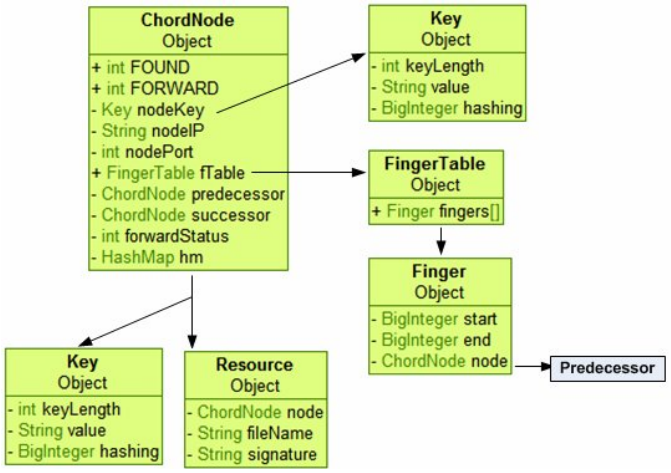

Figure 4.3.2-3: La classe diagramme principale

Nous utilisons un objet *Key* comme index de HashMap. L'objet key contient des objets de type Ressource qui contiennent à leur tour un nœud de type ChordNode (avec l'identité, l'adresse IP, le numéro de port), une liste d'adresses de ressources (par exemple les noms de fichiers, URI), et les signatures de ressources signées par leur propriétaire. Chaque nœud possède un objet *Finger Table* qui stocke les objets *Finger* pour le routage de Chord.

Nous avons choisi de stocker le prédécesseur dans tous les objets *Finger* de la *Finger Table*. Cela réduit les temps de recherche et de publication, mais augmente la taille des objets *Finger* et donc le temps de mise à jour de cette table. L'intérêt de cette proposition a été validé par l'expérimentation (voir 4.3.3 Evaluation expérimentale). Le routage Chord se fait par échange de messages XML en multi-threads de demandes /répondre à/de plusieurs nœuds.

Transport layer Un nœud crée des paires de clés public/privé de manière autonome puis demande à un serveur d'inscription, en tant qu'autorité de certification, de signer son certificat. Un nœud vérifie les clés publiques d'autres nœuds en utilisant un certificat obtenu via le processus de *Handshake* de TLS/SSL. TLS/SSL peut prendre en charge plusieurs combinaisons de chiffrement. Quant à nous nous avons sélectionné l'algorithme TLS_RSA_WITH_RC4_128_MD5 dans notre mise en œuvre car il est très utilisé parmi les organisations financières (par exemple, HSBC, GSB, et Krung Thai) et les serveurs commerciaux (par exemple, Ebay, et Groupama). D'autres arguments pouvant étayér notre choix sont : (1) c'est par défaut la méthode de chiffrement dans la plupart des versions de serveurs fournis par l'éditeur Microsoft (IIS), (2) ces deux algorithmes sont parmi les plus rapides et les moins lourds en termes d'occupation de CPU. De plus, aucune attaque réussie n'a été rapportée à ce jour pour l'utilisation spécifique de la RC4 et MD5 dans SSLv3 avec une clé de 128 bits.

Le processus de négociation de TLS/SSL s'exécute une ou plusieurs fois selon le mode de connexion. Comme RELOAD ne précise pas le mode de connexion, nous avons examiné trois modes de connexion:

1) *Renouveler la connexion à chaque fois*: Nous créons une nouvelle connexion avec tous les nœuds. Elle sera fermée une fois les messages envoyés.
2) *Connexion permanente*: Nous créons une nouvelle connexion avec le nouveau nœud. Elle demeure active tant que l'une des parties n'a pas explicitement demandé sa fermeture.
3) *Connexion temporaire avec timeout*: Nous créons une nouvelle connexion avec le nouveau nœud. Elle reste active tant qu'aucune partie ne l'a fermée et qu'elle n'a pas expirée (timeout).

Banc de test

Notre réseau P2P est composé de 6 nœuds. Il y a 4 ordinateurs, l'un d'eux inclut deux machines virtuelles (avec identifiant 11847 et 50035 à la Figure 4.3.2-4), reliés par un switch Ethernet. Le transfert de message suit le routage de Chord reposant sur les identités. (cf annexe A)

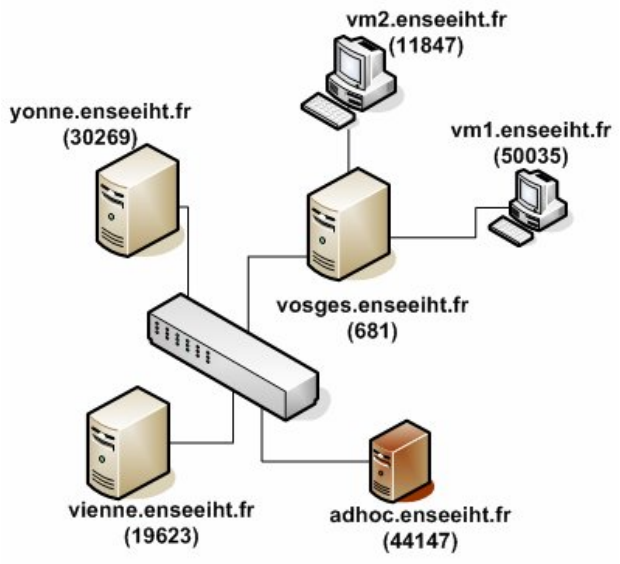

Figure 4.3.2-4: Illustration du banc de test

4.3.3. Evaluation expérimentale

Nous avons utilisé le banc d'expérimentation pour évaluer différents choix de conception.

4.3.3.1. Méthode d'évaluation

Nous avons à partir du banc de test expérimenté plusieurs topologies de réseau pair à pair et avons moyenné les résultats obtenus. Ces résultats sont exprimés en taille de message et en délai ; la taille tient compte du nombre de messages, de leur taille à chaque saut. Pour le délai, nous avons considéré les paramètres suivants:

Distance et topologie :

- Le délai de propagation peut être négligé car de l'ordre de 0,01 sec pour 3.000 km.
- Nombre de saut: même avec notre petit réseau, le nombre de possibilités est trop important pour que tous les cas soient expérimentés. Aussi, nous avons défini 8 façons d'initialiser (définies par l'ordre de connexion des nœuds) le réseau, puis nous avons testé pour chacune six expériences (par exemple, le nœud 1 rejoint le réseau, puis le nœud 2, jusqu'au nœud 6).

Performance des terminaux :

- Dans le cadre du démonstrateur, une capacité de calcul limitée, une carte réseau ou l'exécution d'une machine virtuelle peuvent affecter les résultats. Ceci ne se produira pas dans un déploiement commercial car typiquement, les nœuds seront tous quasi identiques a minima du point de vue matériel.
- Temps d'attente pour une valeur de retour: cette durée (ex, pour demander son successeur) dépend du terminal. Nous utilisons 100 ms.
- L'effet du multi-Threading (java) peut être négligé car il est de l'ordre de 100 microsecondes ou moins.

Choix de conception :

- *Renouveler la connexion à chaque fois:* cette option augmente les délais dans le mode TLS.
- *Connexion permanente:* ce choix peut réduire les délais dans le mode TLS puisque le coût de la négociation TLS est grand par rapport à celui du trafic sans TLS.
- *Structure de données de la Finger table* : il s'agit de stocker ou de ne pas stocker l'élément prédecesseur.

4.3.3.2. Comparaison du délai

Les résultats sont les moyennes de délais mesurés sur 8*6 expériences pour chaque étape de service. Tout d'abord, nous avons mesuré le délai lié à l'étape de *Joining* en choisissant un nœud d'amorçage, dénommé nœud 1. Puis, nous avons mesuré le temps pris par le nœud 2 pour rejoindre le réseau, suivi par le nœud 3, le nœud 4, etc. (il y a donc 6 mesures). Pour la phase de *Publication,* nous avons considéré un fichier photo avec un nom donné (bob.jpg) correspondant à un index stocké sur le nœud même. Puisque le nœud de stockage dépend de l'index, il ne dépend pas du nœud qui publie. Nous avons mesuré le délai pour chaque nœud publiant dans l'étape de publication (nœud 1 au nœud 6). De la même manière, nous avons mesuré le délai dans l'étape de récupération pour chaque nœud. Il est à noté que, de nos travaux antérieurs basés sur le modèle d'analyse et de simulation présentés dans la section précédente [WeFB10][WeFB11] on peut considérer que les formes de résultats sont tout à fait équivalentes pour 1 ou plus publications (1 à 6). Un processus similaire est appliqué pour l'étape *Stabilizing*. Deuxièmement, nous avons fait varier le nœud de démarrage. Finalement, nous avons généré 8 variations qui ont mené à 8 réseaux par l'assignation de différentes identités aux nœuds (1 à 6).

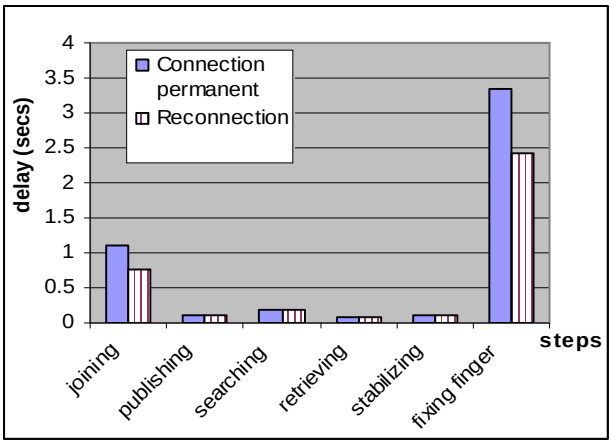

Figure 4.3.3-1: Comparaison des délais des principales étapes de service sur l'architecture P2P/SIP RELOAD selon le mode de connexion

La Figure 4.3.3-1 montre la comparaison entre les délais associés aux modes connexion permanente et reconnexion sans se préoccuper des aspects sécurité. La plupart d'entre eux sont à moins de 0,5 secondes, sauf l'étape de *Joining* et de *Fixing*

Finger, ce qui découle naturellement du nombre d'étapes réalisées durant la phase de *Joining* et par les temps de contacts entre les nœuds dans l'étape de *Fixing Finger*. Pour ces deux étapes on note une différence de délai entre le mode connexion permanente et celui de reconnexion en faveur du mode de reconnexion.

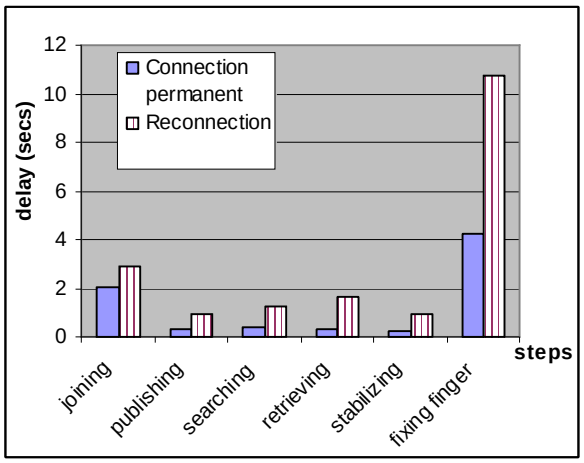

Figure 4.3.3-2 Comparaison des délais des principales étapes de service sur l'architecture P2P/SIP RELOAD avec TLS

Comme la figure précédente, la Figure 4.3.3-2 présente la comparaison entre les délais avec des modes de connexion différents, mais cette fois avec le mécanisme d'authentification qui est TLS. Nous notons alors clairement une augmentation des délai mais également une inversion des rapports de délai selon l'option de connexion. Ici, pour les étapes de *Joining* et *Fixing Finger*, le délai du mode de connexion permanent est plus petit que celui du mode par reconnexion ; ceci car le coût pour trouver un objet de connexion et vérifier son status est moindre que celui pour créer l'objet du TLS/SSL et négocier ses paramètres.

Le délai important de l'étape *Fixing Finger* s'explique par le choix de conception de la structure de données effectué pour la table. Nous avons choisi de stocker le prédécesseur dans un objet *Finger*. Ce choix réduit le temps pour *Publishing* et *Searching* car l'objet *Finger* contient déjà le prédécesseur. Même si l'étape de *Fixing Finger* est plus importante comme elle peut être considérée comme un processus d'arrière-plan, il semble plus intéressant de minimiser le délai des processus de premier-plan en adoptant ce format.

4.3.3.3. Résultat de l'évaluation expérimentale

Les coûts de sécurité dans RELOAD P2P/SIP sont synthétisés sur la Figure 4.3.3-3. Contrairement aux figures précédentes, le ratio des valeurs de la solution sécurisée aux valeurs de la solution non sécurisée plutôt que des valeurs absolues est indiqué. Le coût des messages générés par la sécurité est très faible, sauf pour l'étape de *Joining*. L'étape de *Joining* ne s'exécute pas fréquemment dans la vie des nœuds. Nous pouvons conclure que la solution de sécurité est une solution acceptable en termes de coûts de messages. D'un autre côté, lorsque nous examinons le délai, nous

pouvons remarquer que les mécanismes de sécurité génèrent un délai important dans le mode avec reconnexion.

Ainsi, il serait préférable pour une architecture sécurisée de choisir le mode avec connexion permanente. Cependant, dans le cas où un nœud communique vers tous les nœuds dans l'overlay, le mode connexion permanente peut alors aboutir à une topologie maillé avec un coût d'entretien des liens important. Un nœud devrait pouvoir alors limiter le nombre de connexion simultannées acceptées (en le précisant par exemple dans une configuration).

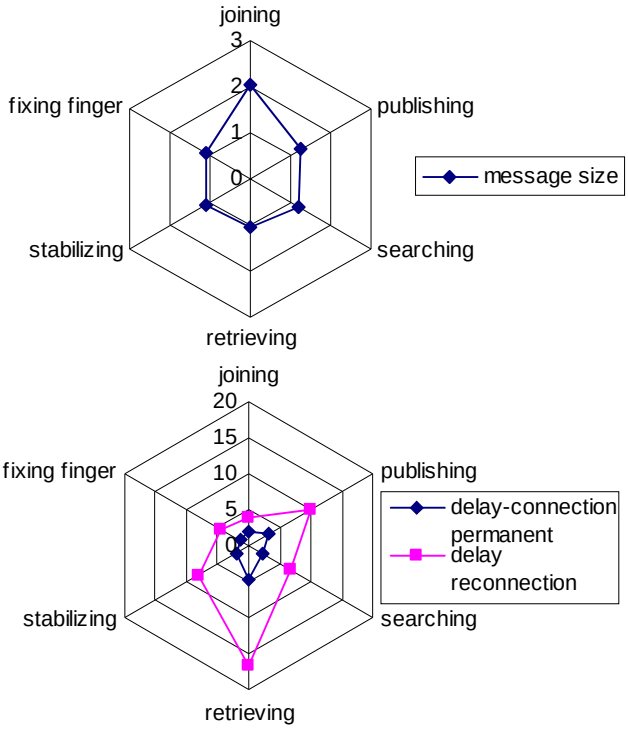

Figure 4.3.3-3: Coût de la sécurité dans P2P/SIP RELOAD

4.3.4. Analyse de signalisation

Evaluons à présent la signalisation de l'architecture que nous proposons.

4.3.4.1. Signalisation comparée de TLS et IBC en DHT P2P

L'expérimentation menée a porté sur le mécanisme TLS qui est le mécanisme de base préconisé dans RELOAD, il nous semble intéressant de prendre également en compte un autre mécanisme de sécurité que nous avons étudié dans la première partie de ce chapitre : le mécanisme IBC.

Nous allons comparer les mécanismes de sécurité de IBC et TLS en DHT P2P. Nos indicateurs de performance sont le coût de transmission et le nombre de signalisation.

La modélisation de ces indicateurs de performance est équivalente à celle effectuée pour l'architecture P2P pur. Nous supposons 1000 nœuds dans le réseau physique (avec $d_{SN\text{-}CN}$ est égal à 7) et de 10 à 100 nœuds (*n*) la taille du réseau overlay (par incrément de 10). Le coût de transmission d'un message de signalisation est, $\gamma = 3{,}84 \times 10^{-4}$ sec.

Analyse des coûts de signalisation

- **DHT P2P avec IBC**

Le coût de signalisation est exprimé selon les phases :

$$C_{total}^{DHT} = C_{register}^{DHT} + C_{publish}^{DHT} + C_{search}^{DHT} + C_{retrieve}^{DHT}$$

Où, chaque phase de transmission de signalisation est donnée par:

$$C_{register}^{DHT} = N{*}R_{reg}\,(\gamma\,(\ (4{*}(\log(n)+3))+7)\ d_{SN\text{-}CN})$$

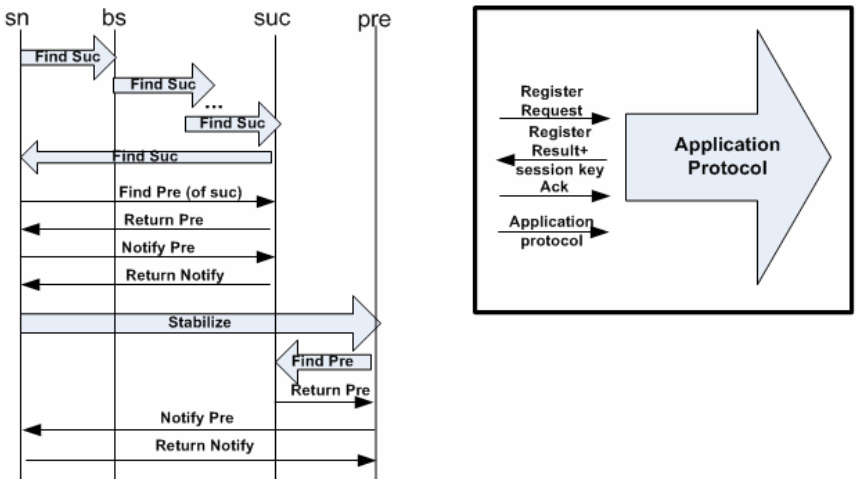

Figure 4.3.4-1: Signalisation de la phase d'enregistrement dans l'architecture DHT P2P avec IBC

La phase d'enregistrement (Figure 4.3.4-1) est composée d'un envoi de l'application *Find successo*r en moyenne à log(*n*) nœuds suivi d'une réponse *Find successor* puis d'un *Stabilize* et enfin un *Find predecessor*. Chacun de ces 4 envois (représenté par une fléche large sur la Figure 4.3.4-1 est en fait constitué de 4 messages de signalisation: 1) *Register Request*, 2) RegisterResult+Session Key, 3) ACK et 4) le contenu applicatif (ex: *Find Successor*) pour établir un canal de communication dans le réseau overlay. Une fois le canal établi un seul message est nécessaire, ainsi 7 messages de signalisations sont échangés entre les nœuds représentés par des flèches fines.

$$C_{publish}^{DHT} = N{*}R_{pub}\,(\gamma\,(\ (4{*}(\log(n)+1))+1)\ d_{SN\text{-}CN})$$

Pour la phase de publication (Figure 4.3.4-2) nous retrouvons un échange multiple (composé de 4 transmissions avec établissements de canal) entre log(*n*) nœuds puis d'un échange direct entre le nœud publiant et le nœud de stockage avec établissement d'un lien suivi du transfert directement sur le lien établi par la source des index à stoker au correspondent CN_x.

$$C_{search}^{DHT} = N*R_{sea}\,(\gamma\,(\quad(4*(\log(n)+1))+2)\,d_{SN\text{-}CN})$$

La phase de recherche (Figure 4.3.4-2) est très similaire à celle de publication, sauf qu'une fois le lien établi entre la source de la recherche et le nœud pair deux messages sont transmis pour rechercher un index.

$$C_{retrieve}^{DHT} = N*R_{ret}(\gamma\,(\,8d_{SN\text{-}CN})\,)$$

Dans la phase de récupération (Figure 4.3.4-2), le lien sécurisé est établi en 3 messages puis une session SIP est mise en place puis terminée en 5 messages (INVITE, OK, ACK, BYE, OK).

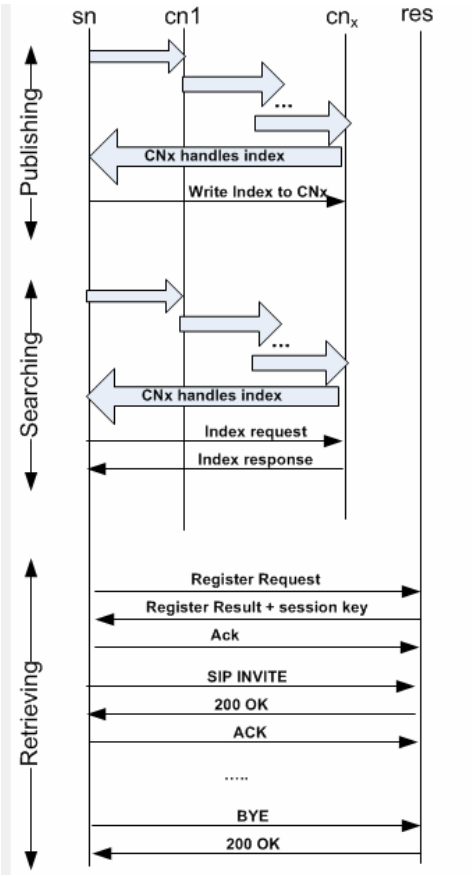

Figure 4.3.4-2: Signalisation de l'architecture DHT P2P avec IBC en phase de publication, de recherche et de récupération

- **DHT P2P avec TLS**

Le fonctionnement avec TLS est très similaire excepté le nombre de messages nécessaires à la création de lien (Figure 4.3.4-3). Alors que 3 messages sont nécessaires en IBC, 5 messages sont utilisés en TLS. Ainsi pour la phase d'enregistrement 6 messages sont échangés pour trouver le successeur avec log(*n*) sauts en TLS (5 messages pour le lien plus le message applicatif : *Find Successor*) alors que 4 messages étaient nécessaires en IBC. (Le handshake de négociation de TLS est plus riche que celui de IBC, il permet de négocier davantage de paramètres)
Soit :

$$C_{total}^{DHT} = C_{register}^{DHT} + C_{publish}^{DHT} + C_{search}^{DHT} + C_{retrieve}^{DHT}$$

Où, chaque phase de transmission de signalisation est donnée par:

$$C_{register}^{DHT} = N*R_{reg}\,(\gamma\,(\ \ (6*(\log(n)+3))+7)\,d_{SN\text{-}CN})$$

$$C_{publish}^{DHT} = N*R_{pub}\,(\gamma\,(\ (6*(\log(n)+1))+1)\,d_{SN\text{-}CN})$$

$$C_{search}^{DHT} = N*R_{sea}\ \ (\gamma\,(\ (6*(\log(n)+1))+2)\,d_{SN\text{-}CN})$$

$$C_{retrieve}^{DHT} = N*R_{ret}(\gamma\,(\,10 d_{SN\text{-}CN})\,)$$

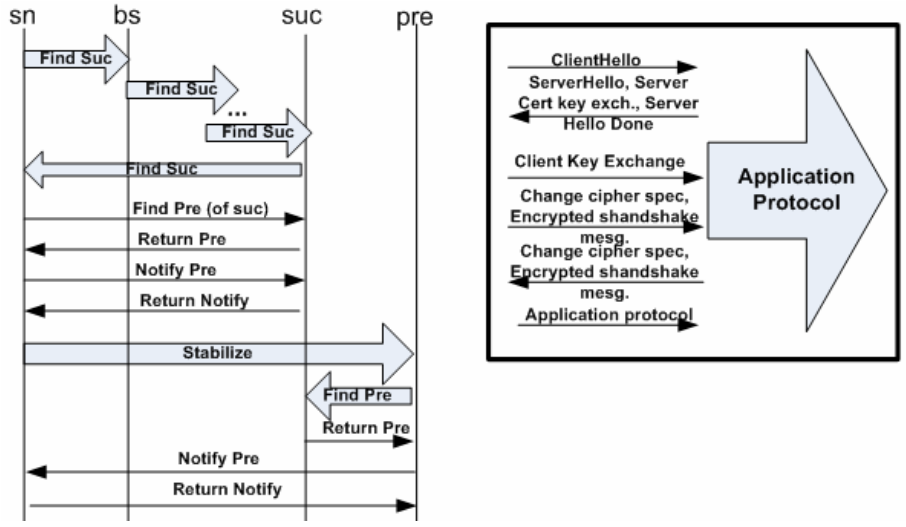

Figure 4.3.4-3: Signalisation de l'enregistrement dans l'architecture DHT P2P avec TLS

Dans la Figure 4.3.4-4, il apparaît naturellement un coût de transmission plus élevé pour TLS que pour IBC, en raison de la phase de négociation. Cependant cet écart n'apparaît pas très important. Une vision plus détaillée des phases est indiquée en Figure 4.3.4-5. Quelle que soit la phase, TLS génère plus de signalisation que IBC.

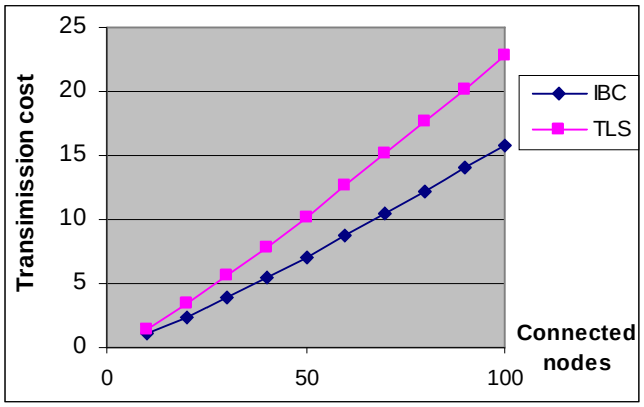

Figure 4.3.4-4: Coût de transmission en DHTP2P IBC et TLS

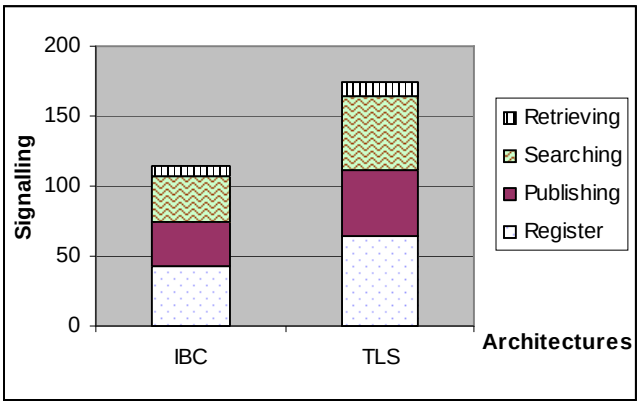

Figure 4.3.4-5: Signalisation IBC et TLS sur l'architecture DHT P2P

4.3.4.2. Comparaison avec les architecture centralisées

Les composantes de la signalisation sont détaillées pour chaque architecture sur la Figure 4.3.4-6. Les signalisations de SIP et d'IMS sont plus faibles que celle du DHT, car il s'agit d'accéder directement à un serveur de catalogue alors que par DHT, il est nécessaire d'acheminer les messages au travers du sur-réseau pour accéder au destinataire en charge de la fonction catalogue.

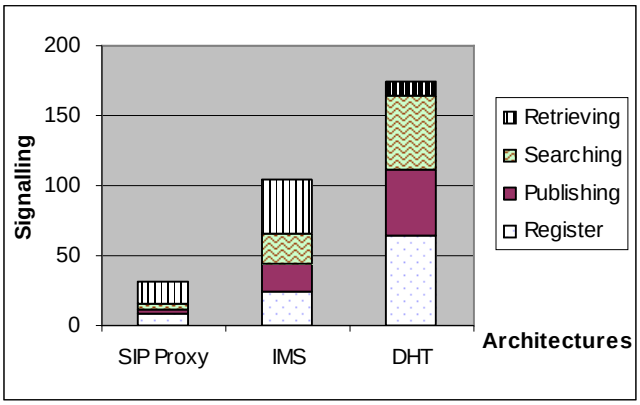

Figure 4.3.4-6: Signalisation comparée dans les architectures de réseau pour une recherche

Nous présentons dans les Figure 4.3.4-7 et Figure 4.3.4-8 la signalisation générée par l'architecture DHT P2P sécurisée avec TLS, préconisé pour RELOAD, pour une seule utilisation du service par session ou bien pour une utilisation multiple (5 recherches). Conformément à la formulation, le nombre de transmission augmente avec le nombre d'utilisateurs actifs. Lorsque le processus est basé sur SIP (en utilisant HTTP Digest et la clé de pre-partagée), le nombre de transmission est minimal. Le P2P DHT (en utilisant TLS) présente le coût le plus élevé, en raison de la recherche du nœud destination dans toutes les phases exception faite de celle de récupération. De plus, le protocole de négociation TLS Handshakes, qui a lieu sur tous les nœuds souhaitant établir une session, augmente encore le nombre de transmission.

Il est à remarquer que plus le service est utilisé plus le coût augmente alors que pour les architectures centralisées, l'utilisation du service ne génère pas le même accroissement.

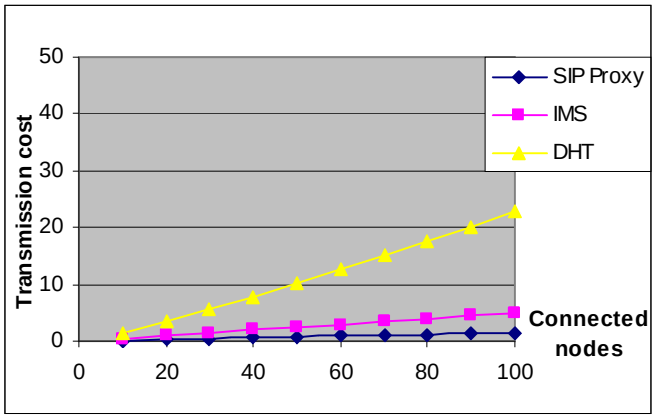

Figure 4.3.4-7: Comparaison des signalisations des architectures avec une seule utilisation de service

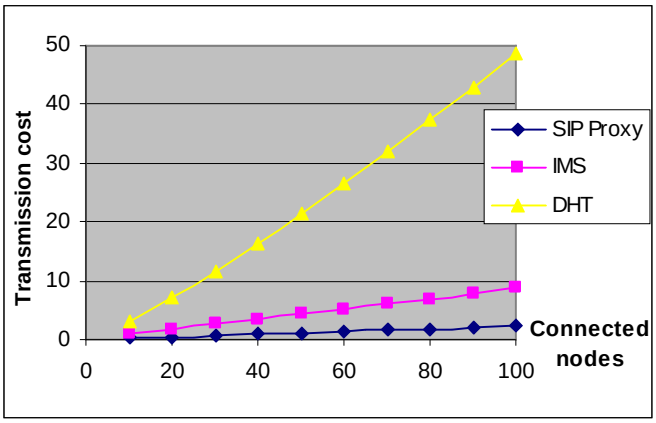

Figure 4.3.4-8: Comparaison des signalisations des architectures avec des recherches multiples

4.4. Conclusion

Dans ce chapitre nous avons étudié les schémas d'architecture en pair à pair distribuée. Nous avons proposé une architecture sécurisée de réseau en pair à pair pur pour les services à domicile et l'avons comparée aux architectures de type centralisé IMS et SIP proxy. Nous avons également considéré une architecture pair à pair distribuée avec un schéma de recherche par table de hachage : DHT P2P.

L'étude des architectures inclut une partie évaluation. Dans les architectures distribuées la topologie du réseau impacte les performances. C'est pourquoi nous avons adjoint à la méthode d'analyse de signalisation employée sur les architectures centralisées, une évaluation par simulation de différentes topologies. L'utilisation de l'outil de génération de topologie BRITE nous permet de donner des valeurs caractéristiques de la topologie que nous utilisons ensuite dans la méthode d'évaluation par analyse de signalisation.

Du point de vue de la sécurité et du déploiement, toutes les architectures considérées présentent certaines similarités, elles utilisent les mêmes éléments de configurations qui peuvent être mis en place grâce à la configuration de la passerelle domestique de l'utilisateur.

Du point de vue performance, l'analyse menée confirme le peu d'intérêt d'une solution basique de pair à pair avec un algorithme de recherche inefficace tel que l'inondation, et indique alors l'intérêt des solutions centralisées.

Pour pallier le défaut de l'algorithme de recherche nous avons proposé le réseau RELOAD en cours d'étude par l'organisme de standardisation Internet. L'expérimentation que nous avons effectuée nous a permis de dégager des choix de conception qui n'avaient pas été précisés par le standard. Bien sûr, l'interprétation des résultats d'expérimentation est limitée, ne reflétant pas les grands réseaux. D'un autre côté, la preuve de concept indique certaines tendances. Ainsi l'utilisation de notre architecture avec une connexion permanente dans le réseau overlay avec TLS/SSL n'a

pas un très grand coût, alors que le mode avec reconnexion est plus coûteux en terme de délai. Nous avons également considéré l'architecture RELOAD avec un mécanisme de sécurité basé sur l'identité. S'il s'avère que l'analyse des échanges de signalisation indique un coût moindre pour la solution IBC, l'écart avec la solution TLS ne parait pas rédhibitoire.

Nous conclurons de cette étude que l'architecture RELOAD semble adaptée à la gestion de services au domicile. Cependant, sa comparaison avec les architectures centralisées indique une moins bonne caractéristique de passage à l'échelle que la solution centralisée, dans la mesure où le coût de signalisation s'accroît avec le nombre d'utilisateurs mais également avec la fréquence d'utilisation du service.

Chapitre 5

5. Conclusion et perspectives

Sommaire

Dans ce document, nous avons présenté les principes des réseaux et services à domiciles appelés à se développer dans le futur avec l'apparition de nouveaux dispositifs communicants performants et ubiquitaires. Puis nous avons introduit différents moyens architecturaux pour délivrer ces services à domicile. Nous nous sommes concentrés sur les architectures pair à pair centralisées et distribuées. Celles-ci n'incluent pas de façon systématique de mécanismes de sécurité ; aussi avons nous ajouté plusieurs mécanismes de sécurité pour protéger les communications. Ils ont été détaillés puis intégrés aux deux types d'architectures centralisées et distribuées avec une illustration du fonctionnement au travers d'un service de partage de photo. Pour pouvoir tester ces mécanismes, nous avons développer deux systémes, un centralisé et l'autre décentralisé. Dans ce dernier chapitre nous concluons ce travail par une synthèse comparative de nos propositions qui s'appuie sur d'autres paramètres que la performance de signalisation examinée dans les chapitres précédents.

Dans un premier temps nous dressons un bilan de notre travail en analysant les avantages et inconvénients des mécanismes élémentaires et plus globalement des architectures de sécurité selon différents points de vue. Nous abordons ensuite l'utilisation par un opérateur et considérons l'aspect gestion et son impact sur l'intérêt d'une architecture.

En second lieu nous abordons les perspectives de notre étude d'abord applicative, en considérant des services plus complexes que celui de partage de photo en mode question/réponse, puis opérationnelle, ainsi que les interactions entre architecture qui doivent être mises en oeuvre pour une extension des services à domiciles qui passe à l'échelle.

5.1. Bilan sur les architectures proposées

Nous avons présenté la délivrance de services à domicile sur des architectures pair à pair pour lesquelles l'utilisateur est le propriétaire des données contrairement à des services délivrés par un serveur qui gère les données de l'utilisateur. Par opposition à une utilisation de l'Internet à partir de serveurs, il s'agit pour l'utilisateur d'exécuter des applications sur son matériel. Trois types de solutions pair à pair ont été étudiées (centralisée, pur et DHT). Leurs principales caractéristiques sont :

Centralized P2P (CP) cette solution que nous avons implantée est la plus simple et la plus aisée à mettre en oeuvre (en comparaison aux autres solutions distribuées). Le goulot d'étranglement crée par des accès nombreux au serveur peut être évité par la mise en place de moyens de traitements importants (nuage de serveurs). Nous avons élaboré une architecture SIP centralisée à laquelle nous avons rajouté des mécanismes de sécurité en s'appuyant sur des éléments de proximité. Nous utilisons le mécanisme soutenu par la norme SIP, HTTP Digest avec clé pré partagée et serveur d'authentification.

Pur P2P (PP) avec cette solution, que nous avons evalué par simulation, l'utilisateur n'a pas à faire confiance à une entité de gestion de ressource car la gestion de partage des ressources est distribuée sur différents nœuds. De plus cette solution évite le

goulot d'étranglement de l'accès au serveur. Pour sécuriser cette architecture, il n'est pas recommandé d'utiliser un serveur d'authentification qui va devoir gérer de nombreux messages, et peut nécessiter également de contacter plusieurs serveurs d'autorités pour vérifier la clé publique en cas de chaînage de certificats. Nous utilisons alors un mécanisme basé sur l'identité. Toutefois, cette technique nécessite de partager des paramètres de sécurité et suppose que la clé privée correspondant à l'identité est sécurisée.

DHT P2P (DP) il s'agit de la dernière génération (actuelle) de pair à pair. Elle est introduite dans le standard, appelée RELOAD, qui est proposé par IETF. Celui-ci recommande également l'utilisation de TLS/SSL pour la sécurité. Nous avons développé une version de RELOAD et testé par expérimentation l'architecture.

5.1.1. Comparaison des mécanismes de sécurité

Nous récapitulons à la suite les caractéristiques des mécanismes :

5.1.1.1. Génération de clé de session

Dans toutes les solutions, une clé de session est utilisée pour sécuriser les communications (plus rapide qu'une clé asymétrique). *Le coût de signalisation minimal est obtenu sur l'architecture PP.*

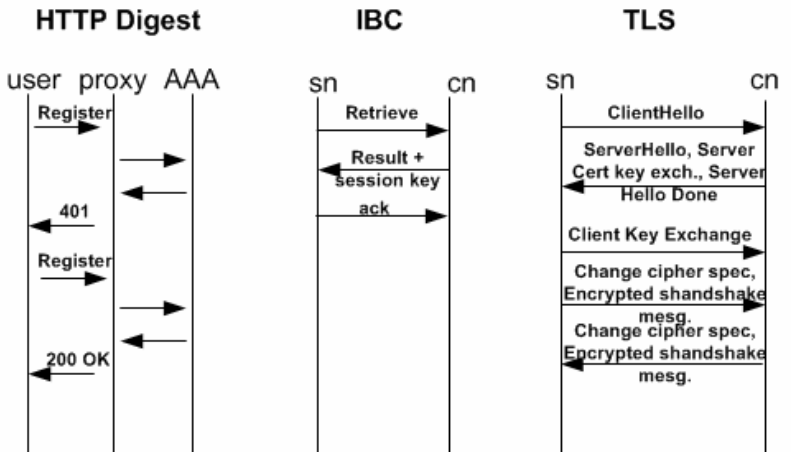

Figure 5.1.1-1: Protocoles des mécanismes HTTP Digest/IBC/TLS pour créer un tunnel sécurisé

Dans l'architecture CP (voir Figure 5.1.1-1) HTTP Digest avec la clé pré-partagée entre un utilisateur et le serveur d'authentification est utilisé pour 2 objectifs (1) authentification mutuelle en faisant un hachage du message avec la clé pré-partagée, (2) pour créer une clé de session entre un utilisateur et le proxy. 8 messsages sont échangés sur le réseau (4 entre l'utilisateur et le proxy et 4 entre le proxy et le serveur d'authentification).
Dans l'architecture PP, nous utilisons IBC pour créer une clé de session entre un utilisateur et son correspondant. Une clé de session est générée à partir du nœud correspondant et transmis au nœud source demandeur. 3 messages sont échangés.

En architecture DP, TLS utilise RSA (algorithme de chiffrement à la clé publique) pour créer une clé de session entre un utilisateur et son correspondant en échangeant 5 messages.

5.1.1.2. Propriétés de sécurité

La comparaison des mécanismes vis à vis des propriétés communes de sécurité [StHF04] que nous établissons est indiquée dans le Table 5.1.1-1. Le symbole 0 signifie que la propriété n'est pas respectée par le mécanisme. Les symboles + et ++ indiquent que le mécanisme marqué ++ vérifie mieux une propriété que celui noté +.

Authentification: Toutes les solutions fournissent une authentification par un hachage MD5 avec des moyens différents. *HTTP Digest* hache un message avec une clé pré-partagée. Des nonces sont également introduits pour éviter les attaques par rejeu, diminuer les collisions. *IBC* hache également un message par MD5 en signant avec une clé privée obtenue par le IBC. TLS fait un hachage MD5 et signe avec une clé privée en utilisant le RSA. *Les trois solutions ont le même niveau en terme d'authentification il s'agit que le correspondant prouve l'identité d'un utilisateur.*

Intégrité: Toutes les solutions assurent l'intégrité du message à l'aide de MD5, soit avec une clé pré partagée soit une clé privée. Ainsi, un nouveau message ne peut pas être forgé. *Les trois solutions ont le même niveau de sécurité : un intrus ne peut pas changer le contenu du message sans savoir une clé pré partagée ou une clé privée.*

Confidentialité: les solutions offrent des niveaux différents en terme de confidentialité. Avec HTTP Digest le message est transmis en clair avec son empreinte, alors qu'avec IBC et TLS qui ont un niveau similaire de confidentialité le message entier est signé et chiffré. *En terme de vie privée, IBC et TLS fournissent plus de confidentialité que HTTP Digest,* dans la mesure où un intrus ne peut pas comprendre la communication.

Non répudiation: *Les solutions IBC et TLS offrent une garantie plus élevée que HTTP Digest,* qui utilise la même clé avec le serveur d'authentification. Toutefois, si le serveur d'authentification est de confiance il n'y a pas de problème, un utilisateur ne peut pas répudier une transaction.

Mise en œuvre: *HTTP Digest est le plus simple à mettre en œuvre* comparé avec IBC et TLS. Il existe de nombreuses bibliothèques de hachage. Par ailleurs, la signalisation est moins complexe que TLS. IBC génère le moins de signalisation, toutefois, elle utilise un algoritme de chiffrement plus complexe pour générer les clés.

Flexibilité*:* *TLS est plus flexible* que HTTP Digest et IBC, car il ne fixe pas tous les mécanismes de chiffrement et d'empreinte du message. Ces paramètres sont négociés au cours des étapes de TLS Handshake. Alors que HTTP Digest et IBC fixent tous les paramètres et algorithmes dans la mise en œuvre.

Performances de vérification de signature: *HTTP Digest peut être considéré comme le plus rapide* car il fait seulement un hachage du message avec la clé pré-

partagée. IBC et TLS utilisent une clé publique pour décrypter une signature puis il la compare au résultat du hachage de messages. Ils ont plus d'étapes de vérification de signature que HTTP Digest n'en a.

Table 5.1.1-1: Mécanismes de sécurité et propriétés de sécurité

	HTTP Digest	IBC	TLS
Authentification	+	+	+
Intégrité	+	+	+
Confidentialité	0	+	+
Non répudiation	+	++	++
Mise en oeuvre	++	+	+
Flexibilité	0	0	++
Performances de vérification de signature	++	0	0
Responsabilité utilisateur	+	0	0

Responsabilité des utilisateurs:

- *Révocation de clé: HTTP Digest avec la clé pré-partagée a un avantage en terme de révocation de clé,* car il suffit de changer la clé pré-partagée avec le serveur d'authentification. Dans IBC et TLS il faut contacter l'autorité de certification afin de révoquer les clés publiques et privées, il faut également prévenir les utilisateurs que la clé publique n'est plus valide.
- *Anonymat: Aucune des solutions ne fournit de garantie sur l'anonymat* de l'utilisateur. Il faudrait rajouter une entité tierce à même d'aveugler certaines informations.

5.1.2. Gestion et dimensionnement du service

Après avoir examiné les propriétés de sécurité des architectures voyons à présent leur adéquation en terme de gestion et dimensionnement de service : quelle gestion pour l'opérateur, pour l'utilisateur, quel impact sur le réseau et le trafic?

5.1.2.1. Mise en place et amorçage

Les trois architectures nécessitent d'utiliser un canal sécurisé préalablement configuré par exemple lors de l'initialisation de la box à son installation au domicile de l'utilisateur. Leur processus d'installation est globalement similaire (Figure 5.1.2-1).

En CP, il s'agit d'installer les proxy, le serveur d'authentification, le catalogue et de créer des tunnels sécurisés entre les éléments AAA, Proxy et Catalogue (si définis sur des sites différents). Puis, il faut générer la clé pré-partagée entre le serveur d'authentification et un utilisateur, et l'envoyer à l'utilisateur sur le canal sécurisé. En PP, il y a une installation d'un serveur PKG avec les paramètres de sécurités. Il faut ensuite demander une clé privée au PKG qui corresponde à l'identité d'un utilisateur (clé publique). Un utilisateur obtient une clé privée et les paramètres de sécurités sur le canal sécurisé. Des nœuds d'amorçage sont également installés afin d'accepter les nœuds entrants. En DP, il faut installer les nœuds d'amorçage ; l'utilisateur génère clé publique/privé/certificat et demande la signature du certificat, auprès de l'autorité de

certification. Il obtient un certificat signé sur le canal sécurisé. L'utilisateur demande également le certificat (contenant la clé publique) de l'autorité afin de pouvoir dans la phase de récupération authentifier la clé publique de son correspondant.

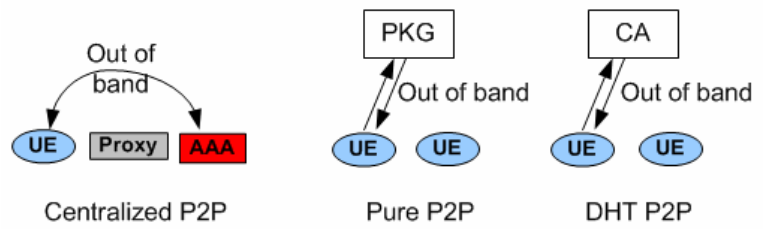

Figure 5.1.2-1: Étape d'amorçage pour les trois architectures

5.1.2.2. Enregistrement

Dans la Figure 5.1.2-2, *La phase d'enregistrement est sécurisée pour toutes les architectures à un coût plus ou moins important, mais le coût n'est pas un facteur décisif* dans la mesure où cette phase n'est pas très fréquente, comme en atteste les résultats du chapitre 5. Cependant les *caractéristiques de passage à l'échelle sont différentes selon les architectures.*

L'architecture CP et l'architecture PP ne sont globalement pas sensibles à l'accroissement du nombre d'utilisateurs qui sont déjà enregistrés (si la capacité du serveur est suffisante). Le coût sera de 2 messages en PP, l'utilisateur devant joindre le nœud d'amorçage qui effectue l'authentification en IBC. En CP il faut établir des tunnels sécurisés entre les différents éléments avec la clé pré-partagée (8 messages).

L'architecture DP est sensible à la taille du réseau. En DP, le client rejoint le sur-réseau avec l'utilisation de TLS pour l'authentification.

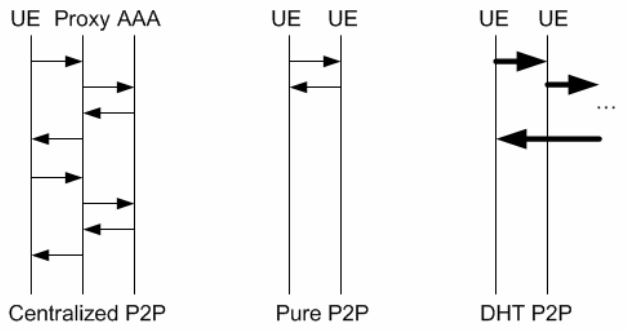

Figure 5.1.2-2: Phase d'enregistrement dans les 3 architectures

5.1.2.3. Publication, recherche récupération et maintenance

La solution PP est la plus consommatrice d'échanges pour la phase de recherche tout en étant la moins consommatrice pour la publication. Cette architecture peut être

envisagée *pour des services de publication asymétriques* où il y a peu de clients qui publient de grandes quantités d'informations (un algorithme de diffusion élaboré devrait permettre de combattre les mauvaises performances relevées dans le chapitre 4) ou bien *pour des petits réseaux* tel un utilisateur souhaitant créer un service familial en raccordant les domiciles.

La solution centralisée (CP) est peu sensible au nombre d'utilisations du service, plus simple et plus rapide sur de grands réseaux que la solution DHT qui est quant à elle sensible au nombre d'utilisateurs enregistrés mais également à la fréquence d'utilisation du service.

Concernant la récupération des données celle-ci s'effectue pour toutes les architectures directement chez l'utilisateur de façon similaire. Le temps d'établissement d'un lien sécurisé est lié au mécanisme mais la différence dans le nombre d'échanges n'est pas significative en terme de délai même si le passage par le proxy pour établir la session SIP peut éventuellement s'avérer plus lent en CP.

Finalement, lorsque l'on considère la phase de *maintenance*, en pair à pair pur et DHT, il est nécessaire d'entretenir la structure du réseau par des échanges de messages entre nœuds ce qui n'est pas le cas en centralisé. En ce sens *la solution CP est plus intéressante*. L'opérateur qui propose une architecture CP a en contrepartie d'un montant élevé de serveur une facilité de gestion. Pour l'utilisateur il y a en CP une utilisation plus faible de la bande passante sur l'ensemble du service. Toutefois, un utilisateur a besoin de faire confiance à un serveur de catalogue qui peut être accédé à partir de tous les utilisateurs authentifiés ; il peut lui être plus facile de faire confiance à un petit cercle d'amis.

5.2. Perspectives

En perspective de notre etude, nous envisageons l'extension de notre analyse à différents services, ainsi que l'étude de l'interconnection d'architectures de services opérées par différents acteurs.

5.2.1. Application à d'autres services

Nous avons au long de ce travail étudié un service simple en question réponse, le partage de photos. Si nous considérons d'autres services, au-delà de l'analyse quantitative, l'intérêt des architectures étudiées s'apprécie alors différemment.

VoIP service
Le service VoIP est un service très populaire qui transmet de la voix ou des vidéos via Internet en mode téléphonique ou téléconférence (par exemple, Skype) à moindre coût. Il peut être exécuté sur toutes les architectures pair à pair que nous avons étudiés. Dans le passé, l'architecture CP a tout d'abord été déployée. L'utilisateur contactait un serveur SIP pour joindre son correspondant connu par son adresse SIP. Quand le nombre d'utilisateurs a considérablement augmenté la solution centralisée a été remplacée par une solution distribuée. Skype, l'application commune de VoIP

utilise une architecture centralisée entre les utilisateurs et un super nœud (un nœud qui a une adresse IP publique et de grandes capacités en termes de bande passante et le traitement), et une connexion entre des super nœuds qui utilise une architecture P2P pur. Un serveur d'authentification centralisé vérifie les utilisateurs. Toutefois l'architecture DHT, avec un index défini simplement à partir de l'adresse SIP, est une solution intéressante. Quand un utilisateur s'inscrit, son identifiant et sa localisation sont déclarés dans le DHT. La recherche se fait simplement sur l'adresse SIP ; un exemple d'un tel système de VoIP est fourni en [BrLo05].

L'opérateur Télécom peut créer son propre réseau Skype avec une *architecture DHT* telle que nous l'avons présentée.

Service de traitement distribué
Ce service est un peu éloigné des services à domicile typiques car il n'y pas de raccordement entre des domiciles utilisateurs, mais le raccordement est entre des domiciles et un « domicile » donné, qui souhaite exécuter un traitement de forte capacité. Il s'agit d'un service qui offre de grandes capacités de traitement en décomposant un traitement en de nombreux petits services et qui les distribue sur un certain nombre d'utilisateurs (par exemple, projet SETI@home, Grid computing). L'objectif est d'utiliser les ressources domestiques de traitement de calcul non utilisées.

Pour ce service, il est plus simple d'avoir une *architecture CP*. La présence d'un point central facilite la vérification et l'assemblage des résultats issus des différents utilisateurs

Service Distribué de streaming
Ceci est semblable à un service de partage de photos (une sorte de service de partage de fichiers), car l'objectif est d'envoyer du contenu numérique à des utilisateurs.

Pour rechercher le contenu multimédia, toutes les architectures peuvent être utilisées. Cependant pour des raisons de responsabilités éditoriales dans la publication de mots clés une architecture CP sera écartée au profit d'une *architecture distribuée DP* chez les clients. L'opérateur ne gère pas d'index, n'est pas responsable de la sémantique des index.

5.2.2. Coopération inter opérateurs

Pour augmenter la capacité de pénétration des services, il faut considérer des déploiements multi opérateurs où chacun propose sa propre architecture pair à pair mais où la continuité de service est garantie à l'utilisateur.

L'hypothèse d'une interaction inter-opérateur repose sur un lien de confiance entre les architectures. Cela signifie, par exemple, que si un utilisateur est authentifié dans son réseau, il peut accéder à un autre réseau qui a un point sécurisé connecté à son réseau.

L'étape d'enregistrement est similaire à celle décrite précédemment, car les utilisateurs inscrivent dans le réseau de leur opérateur ; de même pour la phase de récupération car un lien direct sera établi entre les domiciles s'ils sont de confiance. Pour les étapes de recherche et de publication nous proposons les trois solutions suivantes. La première repose sur une architecture par passerelle, la seconde sur une hiérarchisation de catalogues et finalement nous proposons une structure pair à pair de super nœuds.

- **Solution par passerelle**

Cas de deux architectures CP

L'interopérabilité est illustrée à la Figure 5.2.2-1 qui contient deux éléments passerelle (GW1 et GW2). Le lien entre GW1 et GW2 est un tunnel sécurisé (par exemple, IPSec). 5 rôles principaux sont dévolus à la passerelle. (1) vérifier le message du proxy (2) transmettre un message à une autre passerelle paire, (3) agir comme un utilisateur (par exemple un UE) en émettant des requêtes. (4) attendre le résultat puis retourner celui-ci à la passerelle émettrice et (5) retourner le résultat à un utilisateur d'origine.

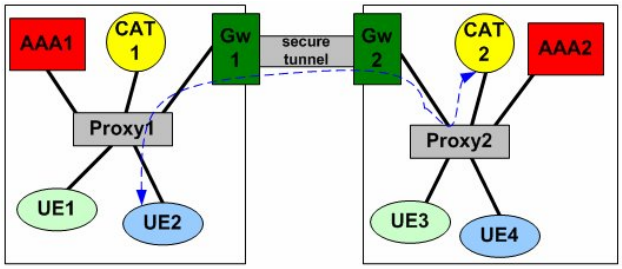

Figure 5.2.2-1: Interopérabilités d'architectures CP par des passerelles.

Les politiques de recherches et publications peuvent être explicites, un UE peut publier dans un autre catalogue en passant par sa passerelle ou rechercher des ressources sur un serveur de catalogue d'un autre opérateur, ou implicite, comme par exemple une politique qui à partir des dernières recherches effectuées par l'utilisateur détermine le catalogue et en cas d'insuccès, dirige la recherche vers un autre catalogue.

Cas d'architectures CP et PP/DP

Dans la Figure 5.2.2-2, le passage d'une requête de recherche entre les deux réseaux nécessite une traduction des messages entre les architectures par un composant adaptateur de message sur les passerelles même si le protocole utilisé est le même (SIP).

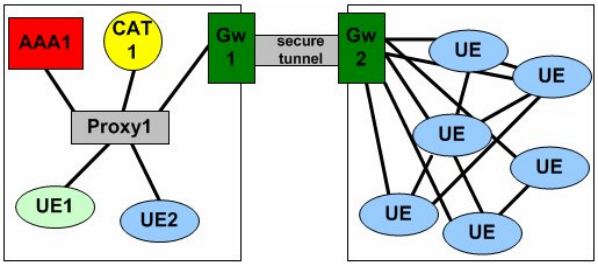

Figure 5.2.2-2: Exemple d'interopérabilité par passerelle

Par exemple, dans la Figure 5.2.2-3, supposons qu'un UE raccordé à une architecture CP recherche un index dans un réseau PP (déterminé par la politique de recherche), UE émet SIP OPTIONS à GW1, GW1 transmet à GW2. GW2 doit créer un message de requête de Recherche à partir de SIP OPTIONS, et émet en inondation dans le réseau PP. Un Id de session. De même, une requête de recherche dans PP est traduite, via SIP OPTIONS, en CP. Tout d'abord, UE envoie une requête de Recherche à GW2. GW2 transmet à GW1. GW1 envoie à CAT1 au travers du Proxy1. Après GW1 obtient des résultats de CAT1 avec SIP 200 OK. GW1 transmet à GW2. GW2 prend les résultats de SIP 200 OK et les retourne à l'utilsateur terminal grâce à l'identifiant de session.

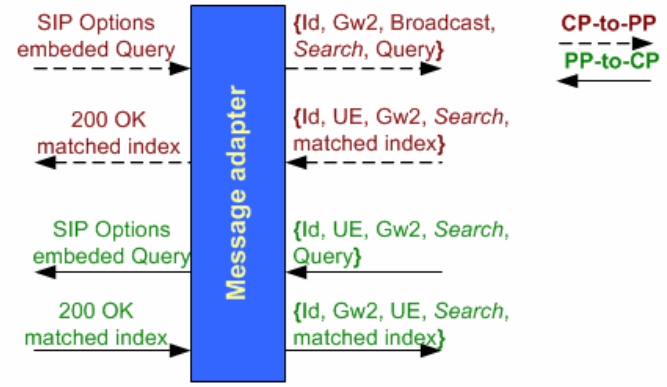

Figure 5.2.2-3: Passerelle de traduction de signalisation pour l'interopérabilité CP/PP.

La solution par passerelle peut être étendue à une architecture DP, cependant dans ce cas il peut être proposé de mettre en œuvre une architecture par catalogue.

- **Solution par catalogue**

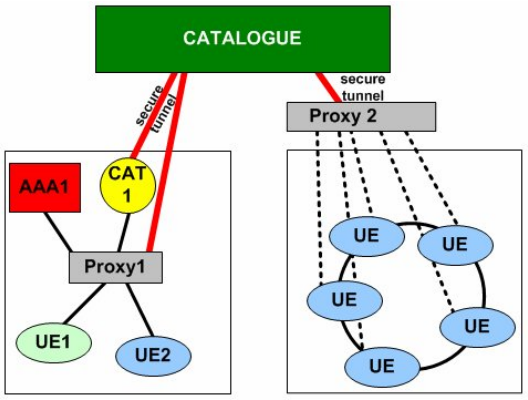

Figure 5.2.2-4: Exemple d'interopérabilité par catalogue

Cette solution utilise un serveur de catalogue centralisé afin de maintenir les index de deux architectures (Figure 5.2.2-4). En CP, l'utilisateur publie sur un catalogue CAT1 local ou directement sur un catalogue central. Les index peuvent contenir un statut pour indiquer s'ils sont relatifs à un serveur local ou seront répliqués sur un serveur global. En DP, UE publie dans un anneau de Chord pour une utilisation locale et dans un catalogue centralisé via Proxy2. Les index peuvent être mis à jour périodiquement ou bien à l'initiative des catalogues locaux. Des tunnels sécurisés permettent de se connecter au catalogue centralisé.

- **Solution par super pairs**

Cette solution utilise des catalogues distribués en tant que super pairs (Figure 5.2.2-5) et raccordés en pair à pair. Une UE publie ou cherche un index localement. Si une UE ne peut pas trouver l'index dans son réseau local, alors elle peut aller directement à un catalogue global (à travers son proxy ou directement). Lorsque la recherche sur le catalogue proche est infructueuse elle est étendue au réseau des catalogues soit en mode inondation du pur P2P soit en mode DHT. Des tunnels sécurisés entre catalogues sont nécessaires. Les index dans les catalogues sont mis à jour par le catalogue local ou périodiquement par des nœuds.

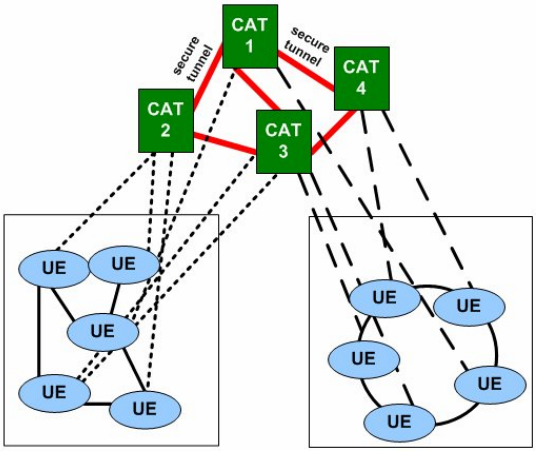

Figure 5.2.2-5: Interopérabilité entre PP et DP par des super pairs

Le choix entre les modes d'interconnexion nécessite d'affiner plus avant la signalisation et les problèmes d'authentification inter-opérateurs (mode de délégation).

Bibliographies

[3GPP12] 3GPP TS 23.228, “IP Multimedia Subsystem (IMS); Stage 2”, Release 11, version 11.4.0, Mar 12

[AdHu00] E. Adar and B. Huberman, Free riding on Gnutella, First Monday, Vol. 5, No. 10, Oct 2000

[ALPH01] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman, “Search in power-law networks”, Physical Review, 2001

[Anan07] S. V. Anand, A DLNA framework for NEXT GEN mobile terminals connecting IMS networks for human-centered Digital Home environment, International Conference on IP Multimedia Subsystem Architecture and Applications, 2007

[AtGM03] S. Ata, Y. Gotoh, and M. Murata, Replication Strategies in Peer-to-Peer Services over Power-law Overlay Networks, APNOMS 2003

[BaAl99] A. L. Barabasi and R. Albert, Emergence of Scaling in Random Networks, Science, pages 509-512, Oct 1999

[BMSW08] D. Bryan, P. Matthews, E. Shim, D. Willis, S. Dawkins, Concepts and Terminology for Peer to Peer SIP, draft-ietf-p2psip-concepts-02, July 08

[BoFr01] D. Boneh, M. Franklin, Identity based encryption from the Weil pairing, 21st Annual International Cryptology Conference, Proceedings of CRYPTO 2001, USA, Aug 2001

[BrLo05] D. A. Bryan and B. B. Lowekamp, SOSIMPLE: A Serverless, Standards-based, P2P SIP Communication System, Advanced Architectures and Algorithms for Internet Delivery and Applications, 2005, June 2005

[BrLZ08] D. Bryan, B. Lowekamp, and M. Zangrilli, The design of a versatile, secure P2PSIP communications architecture for the public Internet, Parallel and Distributed Processing, IPDPS 2008, Miami, Apr 2008

[CaDZ97] K. Calvert, M. Doar, and E. Zegura, Modeling Internet Topology, IEEE Transations on Communications, pages 160-163, Dec 1997

[Cama09] G. Camarillo, Peer-to-Peer (P2P) Architecture: Definition, Taxonomies, Examples, and Applicability, RFC 5694, Nov 2009

[ChAb05] S. Cheshire, B. Aboba, Dynamic Configuration of IPv4 Link-Local Addresses, RFC 3927, IETF Network Working Group, May 2005

[CLZA03] P. Calhoun, J. Loughney, G. Zorn and J. Arkko, Diameter Base Protocol, RFC 3588, IETF Network Working Group, Sep 2003

[CMYP07] X. Chen, K. Makki, K. Yen and N. Pissinou, A new network topology evolution generator based on traffic increase and distribution model, Proceedings of the sixth international conference on networking (ICN’07), Apr 2007

[CoOb] M. D. Collier and M. O'Brien, Hacking VoIP Exposed, http://www.hackingvoip.com/sec_tools.html

[CSFH08] D. Cooper, S. Santesson, S. Farrell, R. Housley, and W. Polk, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, RFC 5280, May 2008

[DiRe08] T. Dierks and E. Rescorla, The Transport Layer Security (TLS) protocol, RFC 5246, Aug 2008

[DLNA] Digital Living Network Alliance, http://www.dlna.org/

[Dong10] C. Dong, JPair project, http://jpair.sourceforge.net/, Oct 2010

[Douc02] J. R. Douceur, The Sybil Attack, Proceedings of 1st International Workshop on Peer-to-Peer Systems (IPTPS), London, 2002

[EJAP01] e-Japan Strategy, Jan 2001, http://www.kantei.go.jp/foreign/it/network/0122full_e.html

[ErRe85] P. Erdos and A. Renyi, Publ. Math. Inst. Hung, Academic Science 5, 17 (1960); B. Bollobas, Random Graphs (Academic Press, London, 1985).

[ETSI11] ETSI TS 187 003 v3.4.1, Technical Specification, Telecommunications and Internet converged Services and Protocols for Advanced networking (TISPAN); NGN Security; Security Architecture, Mar 11

[FaFF99] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On Power-Law Relationships of the Internet Topology, In ACM SIGCOMM, Cambridge, MA, September 1999

[FBHL99] J. Franks, P. Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen and L. Stewart, HTTP Authentication: Basic and Digest Access Authentication, RFC 2617, IETF Network Working Group, June 1999.

[FrBo96] N. Freed and N. Borenstein, Multipurpose Internet Mail Extensions, RFC 2045, IETF Network Working Group, Nov 1996

[Gauc] S. Gauci, SIPvicious SIP scanner, http://www2.packetstormsecurity.org/

[GBPC06] M. Garcia-Martin, M. Belinchon, M. Pallares-Lopez, C. Canales-Valenzuela and K. Tammi, Diameter Session Initiation Protocol (SIP) Application, RFC 4740, Nov 06

[Gnaw02] O. D. Gnawali, A keyword-set search system for Peer-to-Peer Networks, MIT's thesis, Jun 2002

[Guil12] H. Guillaud, Pourquoi Facebook et Google devraient avoir complètement disparu d'ici 5 ans?, Jun 2012, http://blogs.lesechos.fr/internetactu-net/pourquoi-facebook-et-google-devraient-avoir-completement-disparu-d-ici-5-ans-a10806.html

[HaJa98] M. Handley and V. Jacobson, Session Description Protocol (SDP), Apr 1998

[HaRe06] M. Handley and E. Rescorla, Internet Denial-of-Service Considerations, RFC 4732, Nov 06

[JiCJ00] C. Jin, Q. Chen, S. Jamin, On the Placement of Internet Instrumentation, Proc. Of IEEE INFOCOM 2000, Mar 2000

[JLRB12] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset and H. Schulzrinne, REsource LOcation And Discovery (RELOAD) Base Protocol, draft-ietf-p2psip-base-20, Jan 2012

[JoAB01] M. Jovanovic, F. Annexstein, and K. Berman: Modeling peer-to-peer network topologies through "small-world" models and power laws, IX Telecommunications Forum TELFOR 2001, Belgrade

[JoFY07] Y. Joung, C. Fang, L. Yang, Keyword Search in DHT-Based Peer-to-Peer Networks, IEEE Journal on Selected Areas in Communications, pp. 46-61, Issue: 1, Jan 2007

[Jova01] M. A. Jovanovic, Modeling large-scale peer-to-peer networks and a case study of Gnutella, MS. Thesis, University of Cincinnati, Cincinnati, Ohio, USA, 2001

[KeAt98] S. Kent and R. Atkinson, Security Architecture for the Internet Protocol, RFC 2401, IETF Network Working Group, Nov 1998

[KoBu11] M. Kolberg and J. Buford, Application layer multicast extensions to RELOAD, Consumer Communications and Networking Conference (CCNC), 2011 IEEE, Las Vegas, Jan 2011

[KuPo08] A.V. Rajendra Kumar and A. Potluri, Computing segmented backups using resilient routing laters, TENCON 2008, IEEE Region, Nov 2008

[LeFi98] T. B. Lee and R. Fielding, Uniform Resource Identifiers (URI): Generic Syntax, RFC 2396, Aug 98

[LeLD04] B. Leong, B. Liskov, and E. D. Demaine,"EpiChord: Parallelizing the Chord Lookup Algorithm with Reactive Routing State Management", MIT Technical Report MIT-LCS-TR-963, (Cambridge, MA), Aug. 2004.

[LePK06] I. Lee, K. Park and S. Kim, Developments and performance evaluation of digital-home service delivery & management systems, Proceedings of the international conference on Networking, International conference on systems and international conference on mobil

[MaBo10] J. Maenpaa and J. Bolonio, Performance of REsource LOcation and Discovery (RELOAD) on mobile phone, Wireless Communications and Networking Conference (WCNC), 2010 IEEE, Sydney, Apr 2010

[MaDD11] M. Mahdi, O. Dugeon, and M. Diaz, Architectures réseaux pour le partage de contenus multimédias avec garantie de qualité de service, Ph.D. Thesis, Université Paul Sabatier, Toulouse, Nov 2011

[MAGE] T. Magedanz, FOKUS Open IMS Playground, http://www.fokus.fraunhofer.de

[MaMa02] P. Maymounkov and D. Mazières, Kademlia: A Peer-to-peer Information System Based on the XOR Metric. In 1st International Workshop on Peer-to-peer Systems (IPTPS'02), 2002.

[METI03] Ministry of Economy, Trade and Industry, "Society (e-Life strategy society) basic strategy report concerning strategy of making to market of information appliances - e-Life initiative, 2003

[MFMA10] A. Muhammad, P. Fergus, M. Merabti, B. Askwith, Peer-to-Peer overlay gateway services for home automation and management, 24th IEEE international converence on advanced information networking and applications workshops, Apr 2010

[Mili06] C. Militeau, IP Multimedia Subsystem (IMS): Emergency Services, Intrado, ESIF 16, Jan 2006

[MLMB01] A. Medina, A. Lakhina, I. Matta, and J.Byers, BRITE: An Approach to Universal Topology Generation. In Proceedings of MASCOTS '01: The International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, Cincinnati, Ohio,

[MuVW08] A. Munir, W. Vincent and S. Wong, Interworking Architectures for IP Multimedia Subsystems, Computer science mobile networks and applications, Volume 12, Numbers 5-6, Springer Science, Apr 2008

[NFFT08] C. D. Nugent, D. D. Finlay, P. Fiorini, Y. Tsumaki, E. Prassler, Editorial Home Automation as a Means of Independent Living, Automation Science and Engineering, In Automation Science and Engineering, IEEE Transactions Vol. 5, No. 1, pp. 1-9, Jan 2008

[NiAT02] A. Niemi, J. Arkko and V. Torvinen, Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA), RFC 3310, Sep 2002

[PaFW11] B. Paillassa, J. Fasson, W. Werapun, Chapter 8 - Service Management. Dans / In : Digital Home Networking. Romain Carbou, Michel Diaz, Ernesto Exposito, Rodrigo Roman (Eds.), ISTE - WILEY, 8, p. 259-306, 1, 2011.

[PaIf06] V. Pathak and L. Iftode, Byzantine fault tolerant public key authentication in peer-to-peer systems, Computer Networks, Special Issue on Management in Peer-to-Peer Systems: Trust, Reputation and Security, 50(4): 579-596, Mar 2006

[RaBe] M. Ranganathan and J. V. Bemmel, JAIN-SIP Developer Tool, https://jain-sip.dev.java.net/

[RaPa05] M. A. Rahman and A. Pakstas, NeDaSE: A bridge between network topology generator, Network design and simulation tools, The international conference on Computer as a tool, EUROCON 2005, Nov 2005

[ReMo06] E. Rescorla and N. Modadugu, Datagram Transport Layer Security, RFC 4347, Apr 2006

[ReVa02] P. Reynolds and A. Vahdat, Efficient Peer-to-Peer keyword searching technical report 2002, Duke University, CS Department, Feb 2002

[RFHK01] S. Ratnasamy, P. Francis, M. Handley, and R. Karp, A Scalable Content-Addressable Network. In Proceedings of ACM SIGCOMM 2001, Aug 2001

[RoDr01] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In Proceedings of IFIP/ACM Middleware, Heidelberg, Germany, Nov 2001.

[Rose10] J. Rosenberg, Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols, RFC 5245, Apr 2010

[RSCJ02] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. parks, M. Handley and E. Schooler, SIP: Session Initiation Protocol, RFC 3261, IETF Network Working Group, Jun 2002

[RWRS00] C. Rigney, S. Willens, A. Rubens and W. Simpson, Remote Authentication Dial In User Service (RADIUS), RFC 2865, Jun 2000

[SaGG02] S. Saroiu, P. K. Gummadi, and S. D. Gribble, A measurement study of Peer-to-Peer file sharing system, In proceedings of multimedia computing and networking 2002 (MMCN'02), San Jose, USA, Jan 2002

[SCFJ03] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson, A Transport Protocol for Real-Time Applications (RTP) , July 2003

[Scho01] R. Schollmeier, A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications, Peer-to-Peer Computing 2001

[Seed06] J. Seedorf, "Security Challenges for P2P-SIP", IEEE Network Special Issue on Securing Voice over IP, Sep 2006.

[Sham85] Shamir, "Identity-based cryptosystems and signature schemes", Advances in Cryptology (Proceedings of Crypto '84), Lecture Notes in Computer Science, vol 196, Springer-Verllag, 1985

[ShND06] E. Shim, S. Narayanan, G. Daley, An Architecture for Peer-to-Peer Session Initiation Protocol (P2P SIP), draft-shim-sipping-p2p-arch-00, Feb 2006

[SiSc04] K. Singh and H. Schulzrinne, "Peer-to-Peer Internet Telephony using SIP", Columbia University Technical Report CUCS-044-04, , Oct 2004.

[SMKK01] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan Chord: A scalable peer-to-peer lookup service for internet applications. Tech. Rep. TR-819, MIT LCS, March 2001. http://www.pdos.lcs.mit.edu/chord/papers/

[SSSW08] B. Sterman, D. Sadolevsky, D. Schwartz, D. Williams and W. Beck, RADIUS Extension for Digest Authentication, RFC 5090, Feb 2008

[StEB09] M. Steiner, T. En-Najjary, E.W. Biersack, Long Term Study of Peer Behavior in the kad DHT, Networking, IEEE/ACM Transactions on Volume: 17 , Issue: 5, Digital Object Identifier: 10.1109/TNET.2008.2009053, p. 1371 - 1384, Oct 2009

[StHF04] G. Stoneburner, C. Hayden, and A. Feringa, Engineering Principles for Information Technology Security (A Baseline for Achieving Security), Revision A – National Institute of Standard and Technology (NIST) special Publication 800-27 Rev A, Jun 2004

[TaFB06] D. Tarchi, R. Fantacci, M. Bardazzi, Quality of service management in IEEE 802.16 wireless metropolitan area networks, In Proc of IEEE International conference on communication (ICC), Turkey, Jun 2006

[ThSp04] S. A. Theotokis and D. Spinellis, A Survey of Content Distribution Technologies. ACM Computing Surveys, Vol. 36, No. 4, Dec 2004.

[UPnP] Universal Plug and Play Forum, http://www.upnp.org/

[UPnP08] Universal Plug and Play, Device Achitecture 1.1, Document Revision Date: 15 Oct 2008

[WABF09] W. Werapun, A. Abou El Kalam, B. Paillassa and J. Fasson, Solution Analysis for SIP Security Threats, International Conference on Multimedia Computing and Systems, ICMCS 2009, Maroc, Apr 2009

[WaRC00] M. Waldman , A. D. Rubin, and L. F. Cranor, Publius: A robust, tamper-evident, censorship-resistant, In Proc. 9th USENIX Security Symposium, p. 59-72, Aug 2000

[Waxm88] B. Waxman, Routing of Multipoint Connections, IEEE J. Select. Areas Commun., Dec 1988

[WaYu05] X.Wang, K. Yu, How to breack MD5 and other hash finctions-Advancesin Cryptology–Eurocrypt'05, pp.19-35, Springer-Verlag, May 2005

[WeFB10] W. Werapun, J. Fasson and B.Paillassa, Network Architecture for home service delivery, The fourth International Conference on mobile ubiquitous computing, system, services and technologies, UBICOMM 2010, Italy, Oct 2010

[WeFB11] W. Werapun, J. Fasson and B.Paillassa, Home Service Communities and Authentication, Internal Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11, China, Nov 2011

[Weis99] M. Weiser, The computer for the 21st century, ACM SIGMOBILE Mobile Computing and Communications Review, v.3 n.3, p.3-11, July 1999

[WFPM09] W. Werapun, J. Fasson, B. Paillassa, M. Mahdi, O. Dugeon, M. B. Martin, Feel@Home CELTIC Project Deliverable D6.2.1- Extended Home Delivery-Definition & Requirements, Rapport de contrat, D6.2.1, IRIT, September 2009.

[WFPM10] W. Werapun, J. Fasson, B. Paillassa, M. Mohamed, O. Dugeon, M. B. Martin, Feel@Home CELTIC Project Deliverable D6.2.2 Extended Home Delivery Specifications, Rapport de recherche, D6.2.2, IRIT, September 2010

[WuHX08] X. Wu, J. He and F. Xu, An enhanced trust model based on reputation for P2P networks, IEEE International conference on sensor networks, ubiquitous, and trustworthy computing, June 2008

[YHLO03] B. J. You, M. Hwangbo, S. Lee, S. Oh, Y. Do Kwon and S. Lim, Development of a Home Service Robot 'ISSAC', Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas, Nevada, Oct 2003

[YuCS10] Y. Yu, L. Chang and C. Shieh, Increasing service availability for resource constrained nodes in Peer-to-Peer communication systems, Computer Symposium (ICS), 2010 International, Taiwan, Dec 2010

[Zeb06] Z. Zeb, IP Multimedia Subsystem (IMS): Concepts and Architecture, NANC FoN, Telcordia Technologies, Oct 2006

[ZhCY09] Y. Zhang, S. Chen and G. Yang, SFTrust: A double trust metric based trust model in unstructured P2P system, 2009 IEEE International Symposium on Parallel & Distributed Processing, May 2009

[ZhKJ01] B. Y. Zhao, John Kubiatowicz, Anthony Joseph, "Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing", Technical Report, UC Berkeley, 2001

[Zimm95] P. Zimmermann, The Official PGP User's Guide. MIT Press, Cambridge, Massachusetts, 1995

Annexes

A. L'algorithme du Chord P2P

Primitive méthodes

***findSuccessor*:** L'objectif de cette méthode est utilisée pour trouver le successeur. Par exemple (figure A1), N6 voudrais savoir son successeur, il demande un nœud d'amorçage (N0) pour le trouve. Si le nœud (N6) est entre un nœud d'amorçage (N0) et un nœud successeur d'amorçage (N3), alors il peut retourner un successeur (N3), sinon il doit trouver un autre nœud en utilisant la méthode *closePrecedingNode*, et rappelant la méthode *findSuccessor* encore récursivement jusqu'à il trouve son successeur.

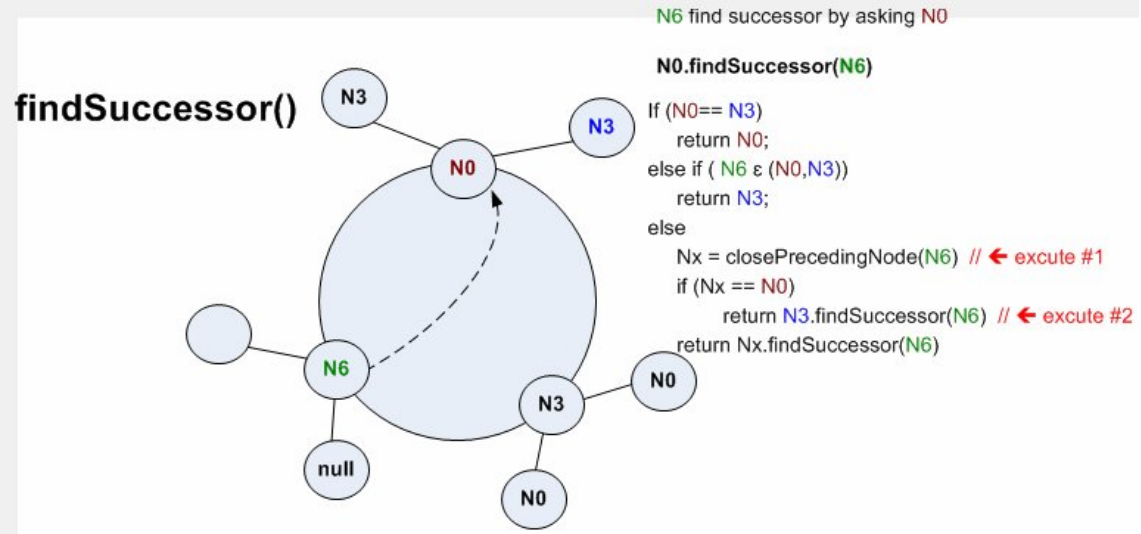

Figure A1: La méthode *findSuccessor* de N6

***closePrecedingNode*:** L'objectif de cette méthode est utilisée pour trouver le nœud qui est proche au nœud d'origine (le nœud qui voudra savoir son successeur). Par exemple (figure A2), il commence par créer une gamme entre les nœuds (N0 - N6). Si un nœud (N0) dans un *finger* élément (repris décroissant) est satisfaite de la gamme, puis il retourne ce nœud (dans un *finger* élément) comme le nœud qui est plus proche au nœud d'origine.

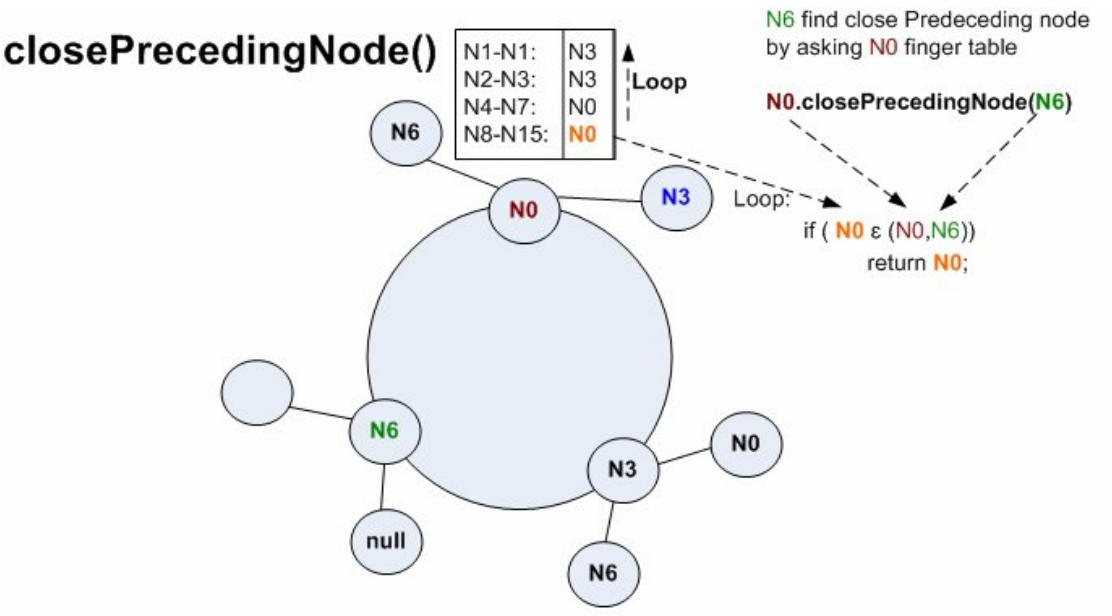

Figure A2: La méthode *closePrecedingNode*

***stabilize*:** L'objectif de cette méthode est utilisée pour mettre à jour le successeur. Par exemple (Figure A3), N3 appel la méthode *stabilize*. Si le prédécesseur (N6) de son successeur (N0) n'est pas en série entre le nœud demandant (N3) et son successeur (N0), puis le prédécesseur (N6) est mis à jour pour le successeur (N0). Puis, cette méthode est appelée la méthode *notifyPredecessor*.

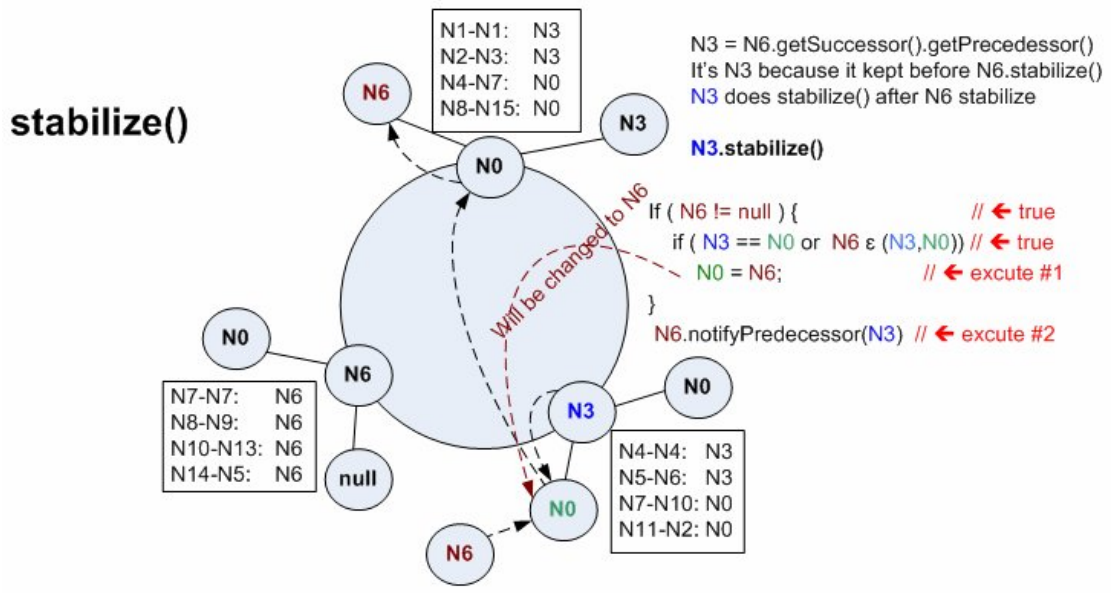

Figure A3: La méthode *stabilize* de N3

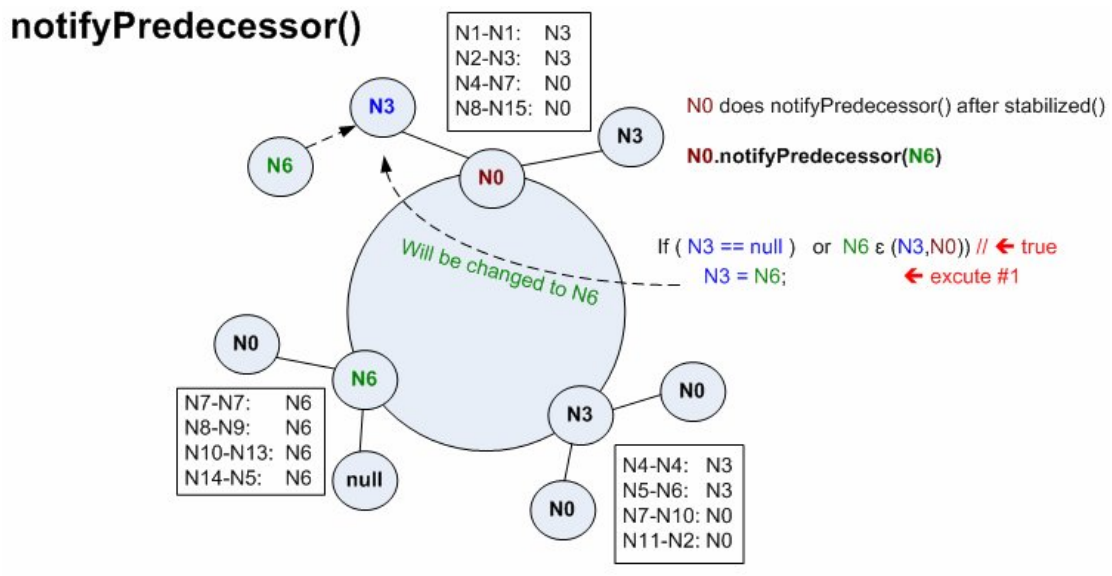

Figure A4: La méthode *notifyPredecessor* de N0

***notifyPredecessor*:** L'objectif de cette méthode est utilisée pour mettre à jour le prédécesseur. Par exemple (Figure A4), N0 appel la méthode *notifyPredecessor*. Si le prédécesseur de (nœud appelant la méthode, N0) est *null* ou si le nœud (nœud paramètre, N6) est entre son prédécesseur (N3) et un nœud d'appel (N0), puis le prédécesseur (N3) est mise à jour par le nœud de paramètre (N6).

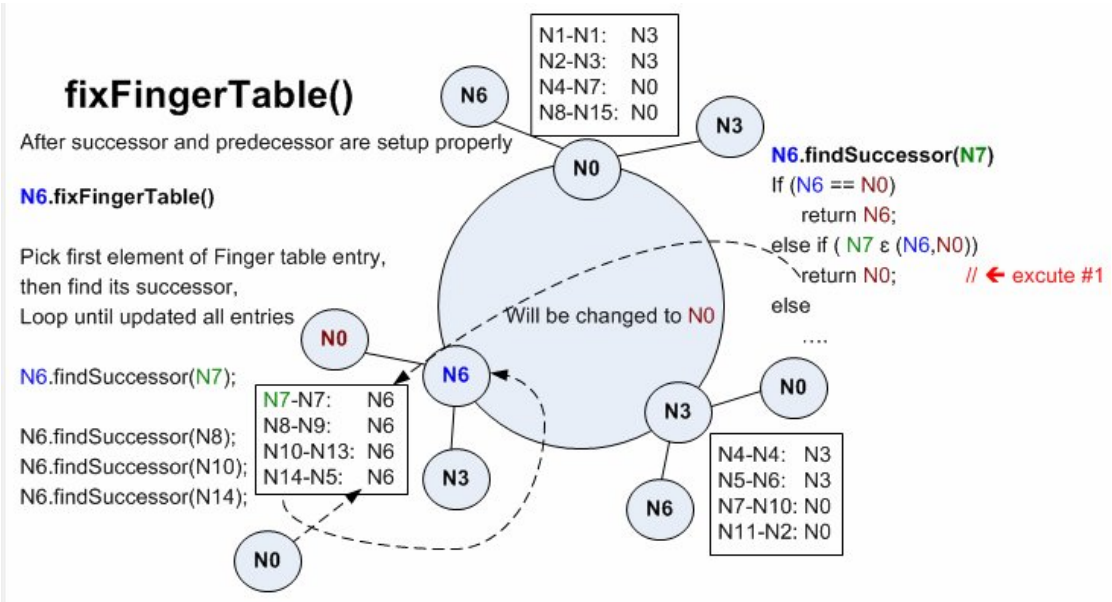

Figure A5: La méthode *fixFingerTable* de N6

***fixFingerTable*:** L'objectif de cette méthode est utilisée pour mettre à jour les nœuds dans *finger* éléments. Par exemple (Figure A5), il prend le premier élément (N7), et de trouver le successeur de N7 (N6). Puis, il va vérifier à l'élément suivant (N6), looping jusqu'à atteindre tous les *finger* éléments de *finger table*.

Etapes de Chord

Pour maintenir la structure overlay et de recherche rapide dans le but de routage et de ajouter d'autres nœuds. Les principales étapes de Chord sont: (1) Initialize ➔ (2) Join ➔ (3) Stabilize ➔ (4) Fix finger table. A l'initialisation, le réseau overlay est vide.

Initialize: Cette étape est appelée une fois que nous créons un nœud, des valeurs initiales pour prédécesseur et successeur doivent mettre à *null* et lui-même respectivement.

Join : Cette étape permet à un nœud de se joindre au réseau overlay existant. Cette étape est appelée une seule fois lorsque le nœud s'ajoute au réseau overlay. Les objectifs de cette étape sont (1) de connaître sa position, (2) de connaître son successeur, (3) de mettre à jour le prédécesseur de son successeur (4) de mettre à jour le successeur de son prédécesseur (5) et de connaître son prédécesseur.
Les objectifs sont apportés par les méthodes suivantes :

(1) par son identité sur l'anneau de Chord.
(2) par la méthode *findSuccessor* récursivement, il peut-être utiliser la méthode *closePrecedingNode* dépendant de son identité.
(3) par la méthode *notifyPredecessor* qui est appelé à partir de la méthode *stabilize* par lui-même.
(4) par la méthode *stabilize* par son predecesseur.
(5) par la méthode *notifyPredecessor* qui est appelé à partir de la méthode *stabilize* par son predecessor.

Stabilize : Cette étape est appelée régulièrement en appelant la méthode *stabilize*.

Fix finger table: Cette étape est appelée régulièrement en appelant la méthode *fixFingerTable*.

Etape Join exemple

Le nœud (N6) se connecte sur le réseau via le nœud amorçage (N0).
Elle s'appuie sur les méthodes :

findSuccesseur (Figure A1)
➔ *closePrecedingNode* (Figure A2)
➔ *findSuccesseur* (Figure A6)
this.stabilize (Figure A7)
➔ *Notify Predecessor* (Figure A4).
predecessor.stabilize (Figure A3)
➔ *Notify Predecessor* (Figure A8)

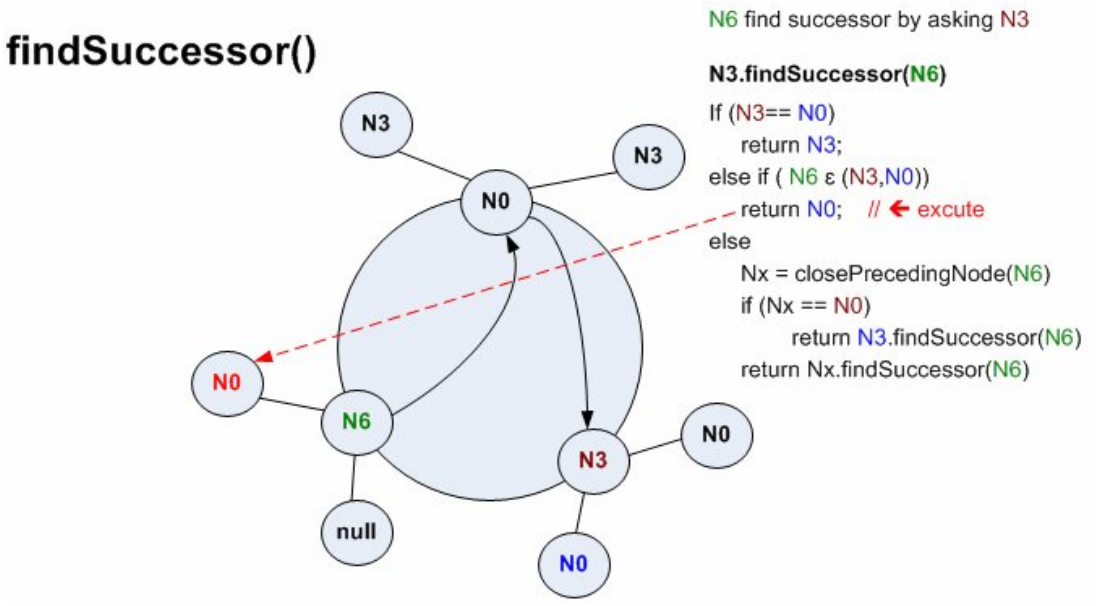

Figure A6: La méthode *findSuccessor* de N3

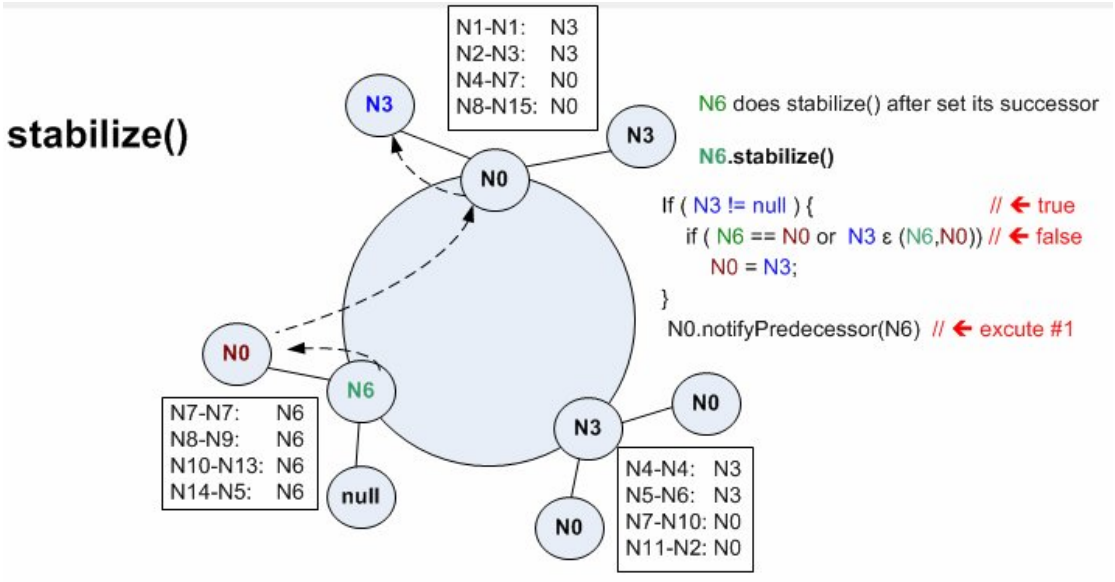

Figure A7: La méthode *stabilize* de N6

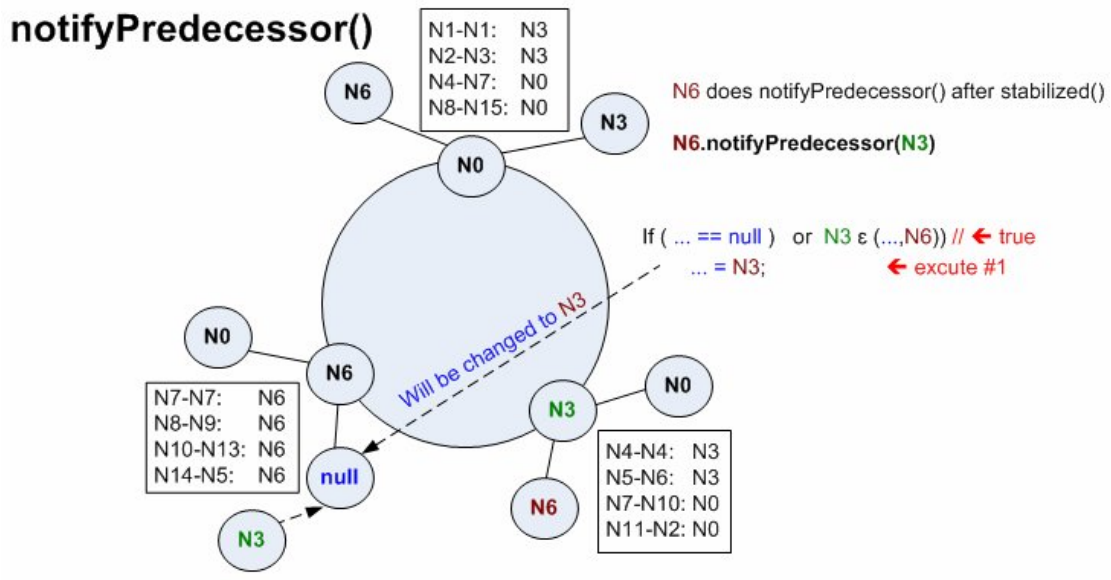

Figure A8: La méthode *notifyPredecessor* de N6

Publications

Conférences et workshops internationaux

- Warodom Werapun, Julien Fasson and Béatrice Paillassa, Home Service Communities and Authentication, Internal Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11, China, Nov 2011
- Warodom Werapun, Julien Fasson and Béatrice Paillassa, Network Architecture for Home Service Delivery, The Fourth International Conference on Mobile Ubiquitous Computing, System, Services and Technologies, UBICOMM 2010, Italy, Oct 2010
- Warodom Werapun, Anan Abou El Kalam, Béatrice Paillassa and Julien Fasson, Solution Analysis for SIP Security Threats, International Conference on Multimedia Computing and Systems, ICMCS 2009, Maroc, Apr 2009

Rapports

- Warodom Werapun, Julien Fasson, Béatrice Paillassa, Mohamed Mahdi, Olivier Dugeon, Marta Bel Martin, Feel@Home CELTIC Project Deliverable D6.2.2 Extended Home Delivery Specifications, Rapport de recherche, D6.2.2, IRIT, Sep 2010
- Warodom Werapun, Julien Fasson, Béatrice Paillassa, Mohamed Mahdi, Olivier Dugeon, Marta Bel Martin, Feel@Home CELTIC Project Deliverable D6.2.1 Extended Home Delivery Definition & Requirements. Rapport de contrat, D6.2.1, IRIT, Sep 2009

Des ouvrages de synthèse

- Béatrice Paillassa, Julien Fasson, Warodom Werapun, Chapter 8 - Service Management, In: Digital Home Networking, Romain Carbou, Michel Diaz, Ernesto Exposito, Rodrigo Roman (Eds.), ISTE-WILEY, 8, p. 259-306, 1, 2011

Résumé : Architectures de réseaux pour la délivrance de services à domicile

Avec l'omniprésence au quotidien du numérique et de l'informatique, de plus en plus d'utilisateurs souhaitent avoir accès à Internet et à leurs applications via n'importe quel périphérique, de n'importe où et n'importe quand. Les appareils domestiques intelligents se développant, les besoins d'échanger des données au domicile même se font de plus en plus sentir. C'est dans ce contexte, celui des services à domicile avec besoin d'interconnexion que se situe notre étude. Ce type de service est qualifié de Home Service (HS) alors que le réseau à domicile est nommé Home Network (HN). La problématique pour les opérateurs est alors de concevoir des architectures appropriées à l'interconnexion des HN de manière sécurisée tout en permettant un déploiement facile et à grande échelle.

Dans la première étape, nous considérons la livraison de services sécurisés à travers un réseau de nouvelle génération (NGN) : IMS (IP Multimedia Subsystem). IMS étant l'architecture de référence pour son caractère réseau NGN des opérateurs, diverses architectures peuvent être développées comme support aux HS. Nous avons choisi d'analyser et de mettre en place une architecture P2P centralisée et de le comparer à l'architecture de référence. Plusieurs mécanismes d'authentification sont mis en place autour du P2P centralisé afin de sécuriser la prestation de services. La modélisation et l'évaluation de notre proposition ont permis d'identifier sa relation à l'IMS mais aussi des problèmes inhérents aux solutions centralisées : la protection des données personnelles, l'impact de la taille sur réseau sur les performances, l'existence d'un point de faiblesse unique face aux attaques et la congestion au niveau du serveur centralisé. Par conséquent, nous nous sommes tournés vers les solutions distribuées pour résoudre ces problèmes.

Dans la deuxième étape, nous considérons l'architecture P2P non-structurée, qualifiée de pur P2P. La cryptographie basée sur l'identité (IBC) est ajoutée au P2P pur afin d'authentifier les utilisateurs et de protéger leurs communications. Pour chacune des solutions une analyse du coût de signalisation est effectuée révélant une faiblesse en ce qui concerne l'étape de recherche. Dans un déploiement à grande échelle, le coût de cette phase est trop élevé. Aussi, nous examinons le P2P structuré basé sur les Dynamic Hash Tables, une autre solution distribuée. Cette architecture est étudiée par l'IETF en tant qu'une des dernières générations de P2P: *REsource LOcation And Discovery (RELOAD) Base Protocol*. Nous proposons son utilisation dans le cadre des HSs. Comme preuve du concept, cette solution a été implantée et déployée sur un petit réseau en utilisant TLS/SSL comme mécanisme de sécurité. Cette plateforme nous a permis d'étudier les délais et les coûts de cette solution.

Pour terminer, un bilan est établi sur toutes les solutions proposées En outre, nous introduisons d'autres types de HS et leurs possibilités de déploiement futur.

Mots-clés: services à domicile, réseaux domestiques, architecture réseaux, pair à pair, IMS, SIP, DHT, authentification, sécurité, TLS/SSL, RELOAD